TUGAS AKHIR

(TP.1703)

Optimisasi Bentuk Kapal untuk Memperkecil Tahanan Gelombang

Oleh :

ZULIS IRAWANTO

NRP : 4884100258

Teknik Perkapalan

Fakultas Teknologi Kelautan

Institut Teknologi Sepuluh Nopember

Surabaya

1994

LEMBAR PENGESAHAN

Surabaya, Agustus 1994

Mengetahui

Dosen Pembimbing Tugas Akhir

Digul Siswanto, M.Sc.

FAKULTAS TEKNOLOGI KELAUTAN ITS

JURUSAN TEKNIK PERKAPALAN

T U G A S - A K H I R .

No.: 19/PT12.FTK.2/M/1993

NOMOR/MATA KULIAH	:	TP.1703 /TUGAS AKHIR.
NAMA MAHASISWA	:	ZULIS IRAWANTO
NOMOR POKOK	:	4884100258
TANGGAL DIBERIKAN TUGAS	:	1 Maret 1993
TANGGAL SELESAI TUGAS	:	24 Desember 1993
DOSEN PEMBIMBING	:	Digul Siswanto, M.Sc

TEMA/URAIAN/DATA-DATA YANG DIBERIKAN :

Judul : OPTIMISASI BENTUK KAPAL UNTUK MEMPERKECIL TAHANAN GELOMBANG.---

---sbj---

Surabaya, 6 April 1993

KETUA JURUSAN TEKNIK PERKAPALAN.

Ir. Soejitno

NIP.: 130532029

Dibuat rangkap 4 :

1. Mahasiswa Ybs.
2. Dekan (mohon dibuatkan SK).
3. Dosen Pembimbing (Merah).
4. Arsip Kajur (Kuning).

ABSTRAK

Persoalan optimisasi bentuk lambung untuk meminimumkan tahanan gelombang dapat diselesaikan dengan metode pemrograman matematika. Sebagai pendekatan terhadap tahanan gelombang digunakan Integral Michel, suatu formula tahanan gelombang yang banyak dipakai terutama di dalam persoalan optimisasi. Dengan menggunakan "tent" function untuk mendekati fungsi lambung kapal, Integral Michell bisa ditransformasikan ke dalam bentuk kuadratik standar. Dengan menggunakan constraints pertidaksamaan linier maka teknik pemrograman kuadratik bisa digunakan untuk mendapatkan variasi bentuk kapal optimum untuk tahanan gelombang minimum. Untuk analisa hasil perhitungan, dipilih kapal series 60 block 60 sebagai kapal pembanding. Secara umum, hasil perhitungan menunjukkan bahwa bentuk lambung optimum adalah lambung dengan bulbous bow dan lambung yang bergelombang.

KATA PENGANTAR

Syukur Alhamdulillah penulis panjatkan ke hadirat Allah SWT karena semata-mata atas kehendak-Nyalah tugas akhir ini dapat penulis selesaikan dengan lancar.

Tugas akhir dengan judul "**OPTIMISASI BENTUK KAPAL UNTUK MEMPERKECIL TAHANAN GELOMBANG**" ini disusun sebagai salah satu syarat untuk menyelesaikan studi di Fakultas Teknologi Kelautan Jurusan Teknik Perkapalan Institut Teknologi Sepuluh Nopember Surabaya, guna melengkapi prasyarat kesarjanaan.

Selanjutnya penulis menyampaikan rasa terima kasih yang teramat dalam kepada :

- Bapak, ibu, kakak dan adikku, yang telah memberi dorongan material dan moral selama penulis menempuh pendidikan di jurusan Teknik Perkapalan FTK-ITS Surabaya, hingga penulis dapat menyelesaikan Tugas Akhir ini dengan baik.
- Bapak Digul Siswanto, M.Sc. selaku dosen pembimbing, atas bimbingan, saran dan instruksi yang telah diberikan dalam menyelesaikan tulisan ini.
- Bapak Ir. Asjhar Imron, M.Sc., MSE, PED selaku kepala laboratorium komputasi FTK-ITS.
- Bapak Ir. Achmad Zubaydi M.Eng. dosen wali dan sekretaris jurusan teknik perkapalan.
- Bapak Ir. Soejitno selaku ketua jurusan teknik perkapalan.

- Bapak Ir. S. Tondo Hartono, selaku dekan Fakultas Teknologi Kelautan ITS Surabaya.
- Seluruh dosen teknik perkapalan, khususnya Bapak Ir. P. Andrianto, Ir. Buyung Faraby, Ir. Budi Santosa dan Ir. Murdijanto, M.Eng.
- Rekan-rekan seperjuangan P-28, atas kekompakan dan kebersamaannya.
- Teman-teman "kelelawar malam" yang selalu menemani penulis di Lab. Komputer.
- Sahabat-sahabat terdekatku yang selalu setia dalam suka dan duka.
- Semua pihak yang turut berjasa di dalam penyusunan tugas akhir ini baik secara langsung maupun tidak langsung.

Penulis menyadari bahwa tentunya tulisan ini masih jauh dari sempurna. Karena itu saran, kritik, serta penelitian lebih lanjut akan sangat dibutuhkan untuk semakin memahami fenomena hidrodinamika.

Meski hanya mampu memberikan setetes air di tengah samudera, penulis berharap semoga tulisan ini bermanfaat bagi kita semua.

Surabaya, Agustus 1994

Penulis

DAFTAR ISI

LEMBAR PENGESAHAN		i
LEMBAR PENUGASAN		ii
ABSTRAK		iii
KATA PENGANTAR		iv
DAFTAR ISI		vi
DAFTAR NOTASI		viii
DAFTAR GAMBAR		x
BAB I	**PENDAHULUAN**	
I.1.	Latar belakang permasalahan	I.1
I.2.	Diskripsi permasalahan	I.2
I.3.	Tujuan	I.4
I.4.	Batasan masalah	I.4
I.5.	Metode penlisan	I.5
BAB II	**TEORI TAHANAN GELOMBANG**	
II.1.	Pengertian tahanan gelombang	II.1
II.2.	Sistim Gelombang kapal	II.2
II.3.	Pengaruh interferensi	II.4
II.4.	Teori perhitungan tahanan gelombang	II.8
BAB III	**PROGRAM KUADRATIK**	
III.1.	Pengertian program kuadratik	III.1
III.2.	Persyaratan Kuhn-Tucker	III.3

III.3. Persoalan complementary III.6

III.4. Algoritma complementary pivot III.7

BAB IV PERUMUSAN PERMASALAHAN KE DALAM PROGRAM KUADRATIK

IV.1. prosedur perhitungan numerik integral Michell IV.1

IV.2. Perumusan program kuadratik IV.9

IV.3. Meminimumkan tahanan gelombang untuk beberapa kecepatan kapal IV.12

IV.4. Perumusan constraints untuk permasalahan IV.13

BAB V ANALISA HASIL PERHITUNGAN

V.1. Validasi program V.1

V.2. Analisa bentuk lambung optimum V.5

BAB VI KESIMPULAN DAN SARAN

VI.1. Kesimpulan VI.1

VI.2. Saran VI.2

DAFTAR PUSTAKA

LAMPIRAN

A. Penjelasan program A.1

B. Flowchart program B.1

C. Listing program C.1

DAFTAR GAMBAR

Gambar 2.1. Bentuk gelombang Kelvin

Gambar 2.2. Skema sistem gelombang haluan dan buritan

Gambar 2.3. Kurva tahanan dengan hump dan hollow

Gambar 2.4. Sistim koordinat kapal

Gambar 4.1. Grid framework

Gambar 4.2. Satu unit tent function

Gambar 4.3. Bagian lambung yang terbentuk dari tent function

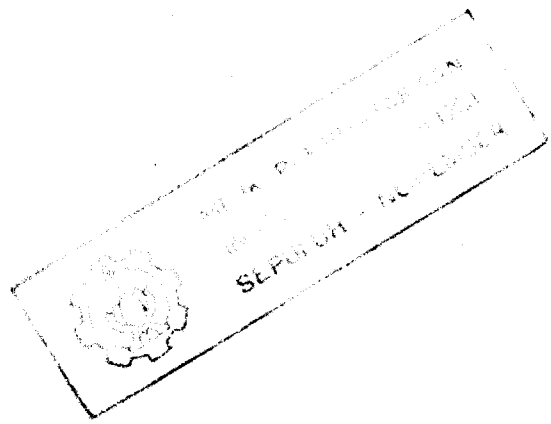
Gambar 5.1. Pengujian hasil optimisasi

Gambar 5.2 sampai 5.19. Rencana garis lambung optimum

Gambar 5.20. Pengaruh bulbous bow terhadap sistim gelombang kapal

DAFTAR NOTASI

A	=	luas penampang
B	=	lebar kapal
C_B	=	koefisien blok
C_m	=	koefisien midship
C_P	=	koefisien prismatik
C_w	=	koefisien tahanan gelombang
C_{wp}	=	koefisien luas garis air
D	=	matrik tahanan gelombang dengan elemen d
d	=	fungsi tahanan gelombang
E_w	=	energi untuk melawan tahanan gelombang
F_n	=	bilangan Froude
g	=	percepatan gravitasi bumi
H	=	fungsi lambung
h	=	fungsi lambung non dimensional
L	=	panjang kapal
L_w	=	panjang gelombang
R_w	=	tahanan gelombang kapal
S	=	jarak
s	=	variabel slack
T	=	sarat kapal
t	=	waktu
V	=	kecepatan kapal



v	=	variabel slack
x	=	posisi non dimensional dalam arah mamanjang kapal
y	=	posisi non dimensional dalam arah melintang kapal
z	=	posisi non dimensional dalam arah vertikal
z_0	=	variabel artificial
γ_0	=	bilangan gelombang nondimensional
ε	=	sudut fase gelombang
ζ	=	posisi dimensional dalam arah vertikal
ζ_w	=	amplitudo gelombang
ζ_Λ	=	tinggi gelombang
η	=	posisi dimensional dalam arah melintang kapal
φ	=	sudut antara garis station dan garis dasar
θ	=	sudut antara garis air dan garis tengah kapal
μ	=	faktor pengali Lagrange
ξ	=	posisi dimensional dalam arah mamanjang kapal
ρ	=	massa jenis air laut
ω	=	frekuensi sudut
∇	=	volume displesemen

BAB I
PENDAHULUAN

1.1. Latar Belakang permasalahan

Di dalam perancangan kapal dagang terdapat 2 hal yang menjadi persyaratan pokok, yaitu kapasitas muatan dan kecepatan kapal. Kapasitas muatan menentukan ukuran utama kapal, sedangkan kecepatan menentukan sistim penggerak kapal.

Sistim penggerak kapal terdiri atas propulsor, power plant, dan lambung kapal. Aspek tenaga penggerak merupakan masalah yang sangat penting di dalam perancangan kapal terutama ditinjau dari segi ekonomis pemakaian bahan bakar. Untuk itu semua elemen di dalam sistim penggerak kapal harus dimainkan seoptimal mungkin sedemikian sehingga jumlah energi yang diperlukan untuk menggerakkan kapal harus sekecil mungkin.

Dari ketiga elemen sistim penggerak kapal, tampaknya perancangan bentuk lambung merupakan pemecahan terbaik dan menarik untuk dikaji. Pemilihan bentuk lambung yang optimal akan menurunkan tahanan kapal yang berarti memperkecil daya mesin penggerak kapal dan dengan demikian akan meningkatkan efisiensi pemakaian bahan bakar.

Seperti diketahui, secara garis besar tahan kapal terdiri atas 2 komponen yang paling dominan, yaitu tahanan gesek dan tahanan gelombang. Tahanan gesek dipengaruhi oleh faktor kekasaran permukaan dan luas permukaan basah. Sehingga pemilihan bentuk lambung tidak

banyak membantu di dalam memperkecil tahanan gesek. Walaupun demikian, perancangan bentuk lambung yang baik akan berpengaruh besar di dalam penurunan tahanan gelombang. Meskipun pada kecepatan rendah besarnya tahan gelombang kurang berarti , namun pengaruhnya semakin besar dengan bertambahnya kecepatan kapal, bahkan bisa mencapai hingga 50 persen dari tahanan total.

Sejak lama masalah ini selalu menarik perhatian para perancang kapal dan para ahli hidrodinamika. Berbagai metode telah dikembangkan mulai dari observasi langsung hingga penggunaan model fisik dan model matematis. Beberapa percobaan yang telah mereka lakukan di towing tank melahirkan berbagai bentuk kapal standar yang dinyatakan sebagai kapal berkualitas tinggi. Dari sana ditemukan pula bentuk-bentuk lambung nonkonvensional, antara lain bulbousbow, yang memiliki beberapa keunggulan di dalam sifat-sifat hidrodinamis kapal.

Penggunaan model matematis semakin banyak dipakai terutama dengan semakin berkembangnya teknologi komputer yang memungkinkan untuk bekerja dengan model matematis berskala besar dan canggih. Dengan memanfaatkan komputer sebagai media pemroses data yang cepat dan akurat maka perancangan bentuk lambung kapal yang optimal dapat dilakukan dengan lebih hemat waktu dan tenaga.

I.1. Diskripsi permasalahan

Bentuk lambung kapal optimum untuk tahanan gelombang minimum bisa didapatkan dengan menggunakan teknik pemrograman matematika, yaitu suatu metode optimisasi untuk mendapatkan nilai optimum dari suatu

fungsi numerik, yang disebut fungsi obyektif, dari sejumlah variabel di dalam beberapa constraints tertentu. Di dalam permasalahan ini, yang menjadi fungsi objektif adalah tahanan gelombang, sedangkan constraint yang digunakan adalah bentuk geometris lambung kapal. Baik fungsi obyektif maupun constraints merupakan fungsi dari offsets lambung kapal *y*. Pemilihan variabel *y* sedemikian sehingga dihasilkan tahanan gelombang minimum adalah jawaban yang akan dicari.

Optimisasi didasarkan pada sebuah kapal pembanding dan dilakukan dengan cara mengoptimumkan bentuk pada bagian-bagian tertentu dari lambung kapal tersebut. Sebagai kapal pembanding dipilih kapal series 60 block 60, karena merupakan kapal dagang standar berkualitas tinggi dan telah banyak dipakai secara luas sebagai acuan di dalam perancangan kapal serta paling ramping diantara kapal-kapal series 60.

Integral Michell dipilih sebagai model matematis untuk perhitungan tahanan gelombang, karena merupakan formula yang sederhana namun memberikan hasil yang cukup memuaskan terutama di dalam permasalahan optimisasi.

Integral Michell merupakan fungsi kuadrat dari offsets lambung kapal, sedangkan semua constraint yang digunakan adalah dalam bentuk linier terhadap offsets lambung kapal. Dengan demikian persoalan optimisasi yang dihadapi adalah persoalan program kuadratik [5].

Dengan merumuskan persoalan optimisasi ke dalam bentuk persoalan program kuadratik standar, maka suatu prosedur penyelesaian yang didasarkan pada metode Complementary Pivot bisa digunakan untuk mendapatkan offsets lambung optimum.

I.3. Tujuan penulisan

Tujuan yang ingin dicapai dari penulisan tugas akhir ini adalah mendapatkan bentuk lambung kapal yang optimum untuk tahanan gelombang minimum. Hasil dari tugas akhir ini diharapkan bisa membantu di dalam pengembangan kapal dagang berkecepatan tinggi di masa mendatang.

I.4. Batasan Masalah

Agar pembahasan di dalam tugas akhir ini tidak meluas dan bisa mengarah kepada tujuan yang hendak dicapai, maka digunakan batasan-batasan sebagai berikut :

- Sesuai dengan judul tugas akhir, pengertian bentuk lambung optimum hanya ditinjau dari segi tahanan gelombang.
- Perhitungan tahanan gelombang menggunakan integral Michell.
- Kapal series 60 block 60 dipakai sebagai kapal pembanding di dalam melakukan analisa.
- Mengingat keterbatasan memory komputer, perhitungan hanya dilakukan untuk lambung dengan 13 station dan 6 garis air.
- Jika terjadi degenerasi (iterasi yang tidak berakhir) selama proses optimisasi, perhitungan tidak dilanjutkan.

I.5. Metodologi penulisan

Metode yang dipakai di dalam penulisan tugas akhir ini adalah:

- **Studi literatur**

  Penulisan tugas akhir berdasarkan literatur-literatur yang mempunyai relevansi dengan permasalahan.

- **Program komputer**

  Permasalahan dirumuskan sacara matematis sehingga mudah untuk dimanipulasi dengan komputer.

- **Analisa hasil**

  Hasil yang didapat dari perhitungan dianalisa berdasarkan teori-teori yang sudah ada sehingga diperoleh kesimpulan dari permasalahan.

BAB II
TEORI TAHANAN GELOMBANG

II.1. Pengertian tahanan gelombang kapal

Tahanan gelombang kapal (wave making resistannce) adalah komponen tahanan kapal yang berhubungan dengan energi yang dikeluarkan untuk menghasilkan gelombang gravitasi [8]. Tahanan gelombang disebabkan oleh tekanan fluida yang bekerja dalam arah normal terhadap lambung kapal.

Jika benda bergerak jauh di bawah permukaan air dengan kecepatan konstan, tidak ada gelombang yang terjadi, tetapi tekanan normal akan bervariasi di sepanjang benda tersebut. Pada fluida nonviscous resultan gaya akibat variasi tekanan sama dengan nol. Tetapi jika benda bergerak pada permukaan air atau di dekat permukaan air, variasi tekanan tersebut menyebabkan gelombang yang menyebabkan berubahnya distribusi tekanan pada seluruh lambung, dan resultan dari gaya ke arah depan dan belakang adalah tahanan gelombang kapal [7].

Pada beberapa bagian lambung perubahan tekanan tersebut manaikkan gaya ke arah belakang, dan lainnya menurunkan, tetapi pengaruh dari seluruhnya adalah sama dengan tahanan yang besarnya sedemikian sehingga energi yang dikeluarkan oleh benda untuk melawan tahanan tersebut sama dengan energi yang dibutuhkan untuk mempertahankan sistim gelombang.

Tahanan gelombang dapat dipisahkan menjadi 2 macam :

- **Wave pattern resistance :**

  yaitu komponen tahanan yang diperoleh dari pengukuran elevasi gelombang, dimana diasumsikan bahwa medan kecepatan di bawah permukaan air, dan momentum fluida dapat dihubungkan dengan bentuk gelombang dengan menggunakan teori linier, sehingga tidak mencakup wave breaking resistance.

- **Wave breaking resistance :**

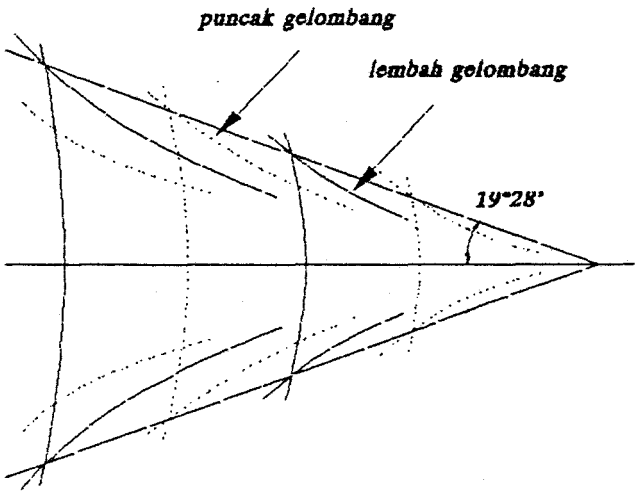
  yaitu komponen tahanan yang berhubungan dengan pemecahan gelombang pada daerah haluan kapal.

Pada umumnya tahanan gelombang diartikan sebagai wave making resistance dengan mengabaikan wave breaking resistance.

II.2. Sistim gelombang kapal

Jika sebuah benda bergerak melewati fluida ideal, maka potensial dan kecepatannya akan berubah dari titik ke titik. Ini berarti bahwa menurut persamaan Bernoulli tekanan akan berubah dari titik ke titik. Oleh karena itu sebuah benda yang bergerak horizontal di sekitar permukaan bebas menyebabkan gangguan pada permukaan tersebut. Variasi tekanan tersebut diwujudkan dengan perubahan ketinggian fluida. Perubahan tersebut akan bergerak dengan kecepatan yang sama dengan benda.

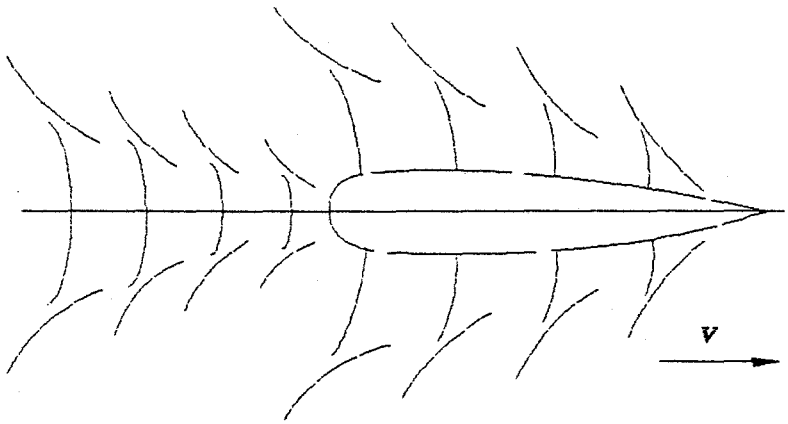
Penelitian tentang masalah ini pertama kali dilakukan oleh Lord Kelvin. Dia menganggap sebuah titik tekanan yang bergerak lurus melewati permukaan fluida, menghasilkan suatu gelombang yang membentuk sebuah



Gambar 2.2. Sitim gelombang Kelvin

pattern. Pattern tersebut terdiri atas sistim gelombang melintang dan sistim gelombang memencar dalam arah radial dari titik tersebut. Seluruh pattern tersebut berada di dalam dua garis lurus yang berawal dari titik tekanan dan membentuk sudut19°28' pada kedua sisi.

Pattern gelombang Kelvin memberi gambaran beberapa sifat dari sistim gelombang kapal. Pada bagian depan kapal terdapat daerah bertekanan tinggi sehingga terjadi gelombang haluan sebagai bagian dari sistim gelombang melintang dan memencar. Garis puncak gelombang melintang akan tegak lurus terhadap arah gerakan di dekat lambung, berbelok ke belakang mendekati sistim geombang memencar sampai akhirnya tidak tampak di dalam sistim gelombang memencar. Sistim gelombang yang serupa juga terjadi pada daerah bahu dan buritan kapal, tetapi tidak dapat dibedakan dengan jelas karena adanya gangguan yang telah terjadi pada sistim gelombang haluan.



Gambar 2.2. Skema sistim gelombang haluan dan buritan

Sistim gelombang kapal dapat dianggap sebagai resultan dari 5 komponen, yaitu :

1. Gangguan simetris pada permukaan, yang memiliki puncak pada daerah haluan dan buritan.
2. Gelombang haluan, yang dimulai dengan puncak gelombang.
3. Gelombang bahu depan, yang dimulai dengan lembah gelombang.
4. Gelombang bahu belakang, yang dimulai dengan lembah gelombang.
5. Gelombang buritan, yang dimulai dengan puncak gelombang.

Komponen pertama disebut sistim gelombang primer, dan empat komponen lainnya di sebut sistim gelombang sekunder.

II.3. Pengaruh interferensi

Kelima komponen dari sistim gelombang akan berinterferensi satu dengan yang lain dan membentuk sistim gelombang total. Dengan naiknya

kecepatan maka panjang gelombang dari masing-masing sistim gelombang sekunder akan naik. Karena puncak dan lembah gelombang pertama tetap pada kedudukannya, profil gelombang total akan berubah secara kontinyu dengan berubahnya kecepatan karena puncak dan lembah dari sistim yang berlainan saling melewati. Pada kecepatan dimana gelombang yang tinggi dihasilkan, tahanan gelombang akan tinggi, dan sebaliknya.

Gelombang seringkali dinyatakan dalam bentuk sinusoidal. Di dalam gelombang sinusoidal, elevasi gelombang ζ_W dapat dinyatakan dengan:

$$\zeta_w = \zeta_A \cos(\omega t + \varepsilon) \tag{2.1}$$

dimana ζ_A adalah amplitudo gelombang, ω adalah frekuensi sudut, t adalah waktu, dan ε menunjukkan sudut fase pada $t = 0$. Keempat komponen sistim gelombang sekunder memiliki kecepatan, panjang gelombang, dan periode yang sama, dan hanya berbeda dalam tinggi dan fase gelombang. Dengan menjumlahkan keempat komponen gelombang, maka didapatkan persamaan untuk resultan gelombang :

$$\begin{aligned}\zeta^2 = {} & \zeta_1^2 + \zeta_2^2 + \zeta_3^2 + \zeta_4^2 + 2\zeta_1\zeta_2 \cos\frac{2\pi l_{1,2}}{L_w} + 2\zeta_1\zeta_3 \cos\frac{2\pi l_{1,3}}{L_w} \\ & + 2\zeta_1\zeta_4 \cos\frac{2\pi l_{1,4}}{L_w} + 2\zeta_2\zeta_3 \cos\frac{2\pi l_{2,3}}{L_w} \\ & + 2\zeta_2\zeta_4 \cos\frac{2\pi l_{2,4}}{L_w} + 2\zeta_3\zeta_4 \cos\frac{2\pi l_{3,4}}{L_w}\end{aligned} \tag{2.2}$$

dimana $l_{n,m}$ adalah jarak antara puncak gelombang pada sistim gelombang melintang n ke puncak gelombang terdekat dari sistim m, yaitu :

$$\frac{l_{n-1,n}}{L_w} = \frac{\varepsilon_{n-1} - \varepsilon_n}{2\pi}$$

Untuk mendapatkan hubungan antara wave making resistance dengan kecepatan, telah dibuat beberapa pendekatan [8]. Sebagai pendekatan pertama, diasumsikan bahwa energi *E* dari gelombang melintang di dalam sistim gelombang Kelvin adalah berbanding lurus dengan lebar gelombang, kuadrat tinggi gelombang, dan panjang gelombang, yang berarti

$$E = C' b \zeta'_w L_w \tag{2.3}$$

Selanjutnya, dapat diasumsikan bahwa lebar gelombang kira-kira berbanding terbalik dengan panjang gelombang L_W , dan berbanding lurus dengan kecepatan gelombang. Ketika kapal bergerak sejauh *x*, jumlah energi yang diperlukan untuk mempertahankan sistim gelombang dapat diungkapkan dengan

$$\begin{aligned} R_w x &= C' b \zeta^2_w L_w \frac{x}{L_w} \\ &= C' b \zeta^2_w x = C V^2 \zeta^2_w x \end{aligned} \tag{2.4}$$

atau

$$R_w = C V^2 \zeta^2_w \tag{2.5}$$

Dengan mensubstitusikan (2.2), diperoleh ungkapan untuk tahanan gelombang :

$$R_W = C V^2 \left(\sum \zeta^2_{w_a} + \sum 2\zeta_m \zeta_a \cos \frac{2\pi l_{m,a}}{L_w} \right) \tag{2.6}$$

Sekarang diasumsikan tinggi gelombang adalah adalah berbanding lurus dengan perbedaan tekanan pada aliran di sekitar badan kapal dan dengan demikian berbanding lurus denganV^2. Maka

$$R_W = C V^6 \left(1 + \sum C_{ma} \cos \frac{2\pi l_{m,a}}{L_w} \right) \tag{2.6}$$

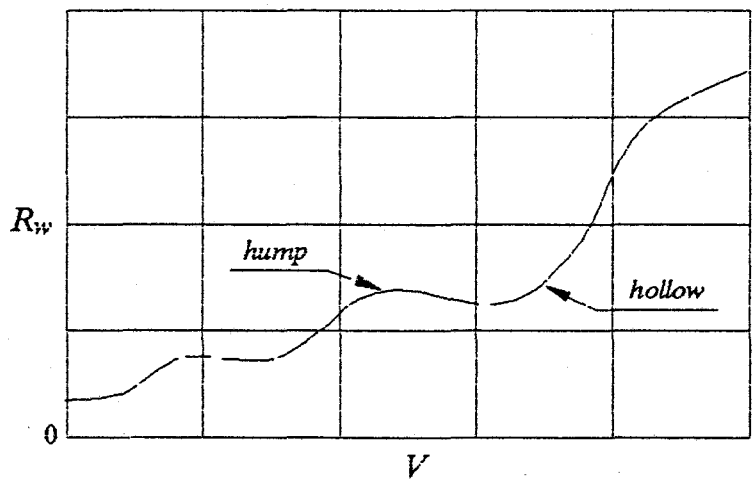
Sehingga koefisien tahanan gelombang C_W adalah

$$C_w = \frac{R_w}{\frac{1}{2}\rho S V^2}$$

$$= C V^4 \left(1 + \sum C_{mn} \cos \frac{2\pi l_{mn}}{L_w} \right) \qquad (2.7)$$

Bentuk CV^4 menunjukkan besar dari wave making resistance jika sistim gelombang individual tidak saling mempengaruhi. Sedangkan bagian terakhir dari persamaan menunjukkan komponen interferensi. Jika bagian ini mempunyai nilai kecil, akan dihasilkan interferensi yang menguntungkan, dan terjadi *hollow* pada kurva tahanan. Sedangkan jika nilainya besar akan terjadi *hump* (gambar 2.3).

Untuk kapal dagang *hump* dan *hollow* umumnya tidak jelas. Akan tetapi, terdapat *hump* pada bilangan Froude 0.30 - 0.35 yaitu pada kecepatan yang sangat tinggi untuk kapal dagang [8].



Gambar 2.3. Kurva tahanan gelombang dengan hump dan hollow

II.4. Teori perhitungan tahanan gelombang

Telah banyak penelitian yang dicurahkan terhadap metode teoritis untuk perhitungan tahanan gelombang. Metode-metode tersebut dapat dibagi dalam 2 kelompok :

- **Metode pertama :**

  Menentukan aliran di sekitar lambung dan kemudian distribusi tekanan normal. Selanjutnya komponen tekanan ke arah depan dan belakang diintegralkan meliputi seluruh permukaan lambung.

- **Metode kedua :**

  Menghitung wave pattern yang dihasilkan kapal pada jarak yang jauh di belakang. Tahanan gelombang selanjutnya ditentukan dari aliran energi yang diperlukan untuk menjaga sistim gelombang.

Metode pertama, yang selanjutnya digunakan dalam penulisan tugas akhir ini, adalah dikembangkan oleh Michell. Metode ini didasarkan pada pendekatan thin ship dan dengan mengabaikan pengaruh viskositas fluida. Michell mendapatkan ungkapan matematis untuk aliran di sekitar slender ship dengan lebar kecil yang berada di aliran uniform. Dari resultan potensial kecepatan, dapat diperoleh distribusi kecepatan dan tekanan pada seluruh lambung, dan dengan mengintegralkan komponen tekanan ke arah depan dan belakang maka dapat diturunkan ungkapan untuk tahanan gelombang total.

Teori sebagaimana dikembangkan Michell adalah valid hanya untuk kondisi tertentu [7] :

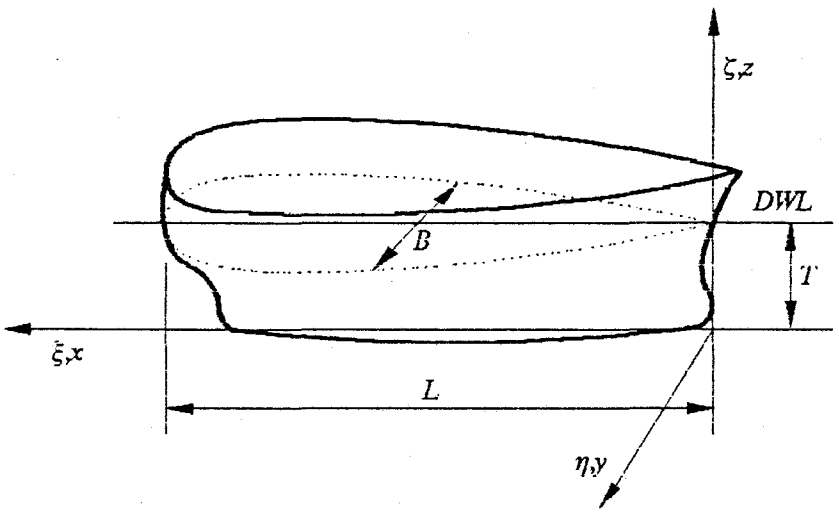
a. Fluida diasumsikan nonviscous dan aliran irotasional. Dalam kondisi ini gerakan dapat ditentukan dengan potensial kecepatan Φ, yang mana harus memenuhi kondisi batas yang diperlukan.
b. Lebar kapal adalah kecil dibandingkan dengan panjangnya ($B/L << 1$), sehingga slope dari permukaan relatif terhadap bidang garis tengah adalah kecil.
c. Tinggi gelombang yang dihasilkan kapal adalah kecil dibandingkan dengan panjangnya, sehingga kuadrat dari kecepatan partikel dapat diabaikan dibandingkan dengan kecepatan kapal.
d. Kapal tidak mengalami singkage atau trim.

Kondisi batas yang harus dipenuhi oleh potensial kecepatan adalah :

a. Pada semua titik pada permukaan lambung, kecepatan normal relatif terhadap lambung harus nol.
b. Tekanan di segala tempat pada permukaan bebas air harus konstan dan harus sama dengan tekanan atmosfir.

Agar masalah tersebut dapat diterima oleh metode matematis yang ada, Michell berasumsi bahwa kondisi batas pertama dapat diterapkan pada bidang garis tengah, sehingga hasilnya bisa dipakai secara tepat pada thin ship, dan bahwa kondisi tekanan konstan bisa dipakai pada permukaan bebas air, dalam kondisi rata dimana distorsi permukaan akibat pattern gelombang diabaikan.

Michell menyajikan tahanan gelombang dalam bentuk integral yang kemudian disebut integral Michell. Integral Michell merupakan integral lipat lima dimana integran terdiri dari fungsi yang menggambarkan bentuk kapal. Untuk sebuah kapal dengan sistim koordinat seperti pada gambar (2.4),



Gambar 2.4. Sistim koordinat kapal

dapat dituliskan integral Michell untuk tahanan gelombang dari thin ship pada kecepatan *V* dan sarat *T* dalam bentuk seperti berikut [1] :

$$R_W = \frac{4\rho g^2}{\pi V^2} \int_1^\infty \frac{\lambda^2}{(\lambda^2 - 1)^{1/2}} \left[I^2(\lambda) + J^2(\lambda) \right] d\lambda \qquad (2.8)$$

dimana

$$\begin{matrix} I(\lambda) \\ J(\lambda) \end{matrix} = \int_0^T \exp\left[\frac{g}{V^2} \lambda^2 (\zeta - T) \right] \cdot \left[\int_0^L H_\xi(\xi, \zeta) \begin{matrix} \cos \\ \sin \end{matrix} \left(\frac{g}{V^2} \lambda \xi \right) d\xi \right] d\zeta$$

ρ = massa jenis air laut

g = percepatan gravitasi bumi

Sejak lama integral Michell digunakan secara luas di dalam persoalan tahanan gelombang. Integral Michell menarik di dalam aplikasi optimisasi

karena beberapa alasan berikut [4] :

- Perhitungan dapat dilakukan dengan relativ mudah dan cepat;
- Secara kuantitatif, tahanan gelombang tidak dihitung dengan cukup akurat (walaupun di dalam pengembangannya adalah mungkin dengan memasukkan koreksi empiris), tetapi prediksi kualitatif dari efek modifikasi bentuk lambung terhadap tahanan gelombang adalah cukup memuaskan;
- Teori tahanan gelombang yang lebih canggih, yang memakan lebih banyak waktu, tidak memberikan hasil yang lebih baik. Sebagai contoh hasil numerik dan eksperimen menunjukkan bahwa bentuk lambung optimum yang didapat dengan menggunakan integral Michell adalah lebih baik dari pada bentuk yang didapat dengan teori low speed baik di dalam pemakaian waktu komputasi maupun di dalam pengurangan tahanan

BAB III

PROGRAM KUADRATIK

III.1. Pengertian program kuadratik

Pemrograman kuadratik merupakan salah satu cabang dari teknik pemrograman matematika, yaitu suatu metode untuk mendapatkan nilai optimum (maksimum atau minimum) dari suatu fungsi numerik, yang disebut fungsi obyektif, dari sejumlah variabel dengan beberapa constraints untuk variabel tersebut. Program kuadratik ditandai dengan fungsi obyektif berbentuk kuadrat dan constraint berbentuk linier. Program kuadratik biasanya dinyatakan dalam bentuk sebagai berikut :

Minimumkan :

$$f(x) = cx + x^{T}Qx \tag{3.1}$$

Dengan constraints :

$$Ax \geq b \tag{3.2}$$

$$x \geq 0 \tag{3.3}$$

$$c = (c_1, c_2, \ldots, c_n) \qquad b = (b_1, b_2, \ldots, b_n)$$

$$x = (x_1, x_2, \ldots, x_n)$$

$$Q = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1n} \\ q_{21} & q_{22} & \cdots & q_{2n} \\ . & . & \cdots & . \\ . & . & \cdots & . \\ q_{n1} & q_{n2} & \cdots & q_{nn} \end{bmatrix} \qquad A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ . & . & \cdots & . \\ . & . & \cdots & . \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

Fungsi obyektif (persamaan 3.1), yaitu fungsi yang dicari nilai optimumnya, dinyatakan dalam bentuk kuadrat, sehingga jenis persoalan ini dinamakan pemrograman kuadratik. Daerah penyelesaian layak didefinisikan oleh constraints pertidaksamaan linier (3.2) dan constraints ketidaknegatipan (3.3) dari variabel x.

Di dalam persoalan optimisasi, optimum sering diartikan sebagai minimum karena maksimum dari suatu fungsi $f(x)$ bisa diperoleh dengan mencari minimum dari negatip fungsi tersebut, $-f(x)$. Suatu fungsi $f(x)$ mempunyai harga minimum global pada titik $\dot{x}$ jika $f(\dot{x}) \leq f(x)$ untuk semua x pada daerah layak. Sedangkan fungsi $f(x)$ dikatakan mempunyai harga minimum lokal pada titik $\dot{x}$ jika $f(\dot{x}) \leq f(\dot{x} \pm h)$ untuk semua nilai h yang kecil.

Di dalam persamaan (3.1), apabila $x^T Q x > 0$ untuk semua x kecuali untuk $x = 0$ maka matrik Q dinamakan positip definit. Sedangkan apabila $x^T Q x \geq 0$ untuk semua x dan sekurang-kurangnya terdapat satu nilai x dimana $x \neq 0$ sehingga $x^T Q x = 0$ maka matrik Q dinamakan positip semidefinit. Jika matrik Q adalah positip definit atau positip semi definit maka fungsi obyektif adalah konvek. Apabila fungsi obyektif adalah konvek dan constraints adalah linier, maka persoalan tersebut adalah persoalan pemrograman konvek. Untuk persoalan pemrograman konvek, setiap penyelesaian minimum lokal yang ada juga merupakan penyelesaian minimum global. Jadi, untuk memastikan konvergensi ke arah minimum global, elemen dari Q harus memenuhi kondisi di atas. Sedangkan elemen c, A, dan b adalah bebas.

Saat ini tidak ada metode yang secara komputasional efisien untuk mendapatkan penyelesaian optimal (global) untuk persoalan pemrograman kuadratik jika Q adalah matrik simetris umum. Di dalam persoalan umum, beberapa algoritma mungkin konvergen ke arah maksimum lokal, minimum lokal, atau bahkan titik belok. Untuk selanjutnya diasumsikan Q adalah positip (semi)definit.

III.2. Persyaratan Khun-Tucker

Di dalam persoalan pemrograman konvek, apabila persyaratan Kuhn-Tucker diterapkan pada program tersebut, maka variabel-variabel penyelesaian yang memenuhi persyaratan Kuhn-Tucker pasti merupakan penyelesaian minimum global. Oleh karena itu dikembangkan sebuah algoritma yang mampu untuk mendapatkan penyelesaian terhadap persyaratan Kuhn-Tucker yang diperoleh dari persoalan pemrograman kuadratik dari persamaan (3.1) sampai (3.3).

Untuk lebih jelasnya, anggaplah persamaan aljabar berikut mewakili persamaan di atas. (Tanpa mengurangi pengertian umumnya, diasumsikan q adalah simetri.)

Minimumkan :

$$f(x) = \sum_{j=1}^{n} c_j x_j + \sum_{j=1}^{n} \sum_{k=1}^{n} x_j q_{jk} x_j \qquad (3.4)$$

Dengan constraints :

$$g_i(x) = \sum_{j=1}^{n} a_{ij} x_j \geq b_i \qquad i = 1, 2, \ldots, m \qquad (3.5)$$

$$x \geq 0$$

Selanjutnya didefinisikan fungsi Lagrange untuk persamaan di atas :

$$L(x,u) = \sum_{j=1}^{n} c_j x_j + \sum_{j=1}^{n} \sum_{k=1}^{n} x_j q_{jk} x_k - \sum_{i=1}^{m} \mu_i \left[\sum_{j=1}^{n} (a_{ij} x_j - b_i) \right]$$

dimana μ adalah faktor pengali Lagrange.

Persyaratan Kuhn-Tucker untuk persoalan di atas didefinisikan oleh persamaan A sampai F.

A. $x \geq 0$

B. $$x_j \left[c_j + 2 \sum_{k=1}^{n} q_{jk} x_k - \sum_{i=1}^{m} \mu_i a_{ij} \right] = 0 \qquad (3.6)$$

C. $$c_j + \sum_{k=1}^{n} q_{jk} x_k - \sum_{i=1}^{m} \mu_i a_{ij} \geq 0 \qquad j = 1, 2, \ldots, n \qquad (3.7)$$

D. $\mu_i \geq 0$

E. $$\mu_j \left[\sum_{j=1}^{n} a_{ij} x_j - b_i \right] = 0 \qquad i = 1, 2, \ldots, m \qquad (3.8)$$

F. $$\sum_{j=1}^{n} a_{ij} x_j - b_i = 0 \qquad i = 1, 2, \ldots, m \qquad (3.9)$$

Misalkan variabel nonnegatip v_j dan s_j berturut-turut menyatakan kuantitas yang berada di dalam kurung pada persamaan (3.6) dan (3.9), maka persamaan tersebut berubah menjadi :

$$\mu_j \left[\sum_{j-1}^{n} a_{ij} x_j - b_i \right] = \mu_j s_j = 0 \qquad i = 1, 2, \ldots, m \qquad (3.10)$$

$$v_j \left[x_j \right] = 0 \qquad j = 1, 2, \ldots, n \qquad (3.11)$$

Persamaan (3.8) menyatakan bahwa *m* constraint asal harus terpenuhi untuk mendapatkan suatu penyelesaian optimal. sedangkan persamaan (3.10)

menyatakan bahwa baik μ_j maupun s_j atau keduanya harus nol pada kondisi optimal. Kondisi ini dikenal sebagai complementary slackness.

Sebagai rangkuman, persyaratan Kuhn-Tucker adalah sebagai berikut :

$$2\sum_{k=1}^{n} q_{jk}x_k - \sum_{i=1}^{m} \mu_i a_{ij} - v_j = -c_j \qquad j = 1, 2, \ldots, n \tag{3.12}$$

$$\sum_{j=1}^{n} a_{ij}x_j - s_i = b_i \qquad i = 1, 2, \ldots, m \tag{3.13}$$

$$\mu_i s_i = 0 \qquad i = 1, 2, \ldots, m \tag{3.14}$$

$$v_j x_j = 0 \qquad j = 1, 2, \ldots, n \tag{3.15}$$

$$x \geq 0 \qquad \mu \geq 0 \qquad v \geq 0 \qquad s \geq 0$$

Sekarang terdapat $(m + n)$ persamaan nonlinier tambahan yang akan menghasilkan sebuah titik stasioner terhadap program kuadratik asal. Di dalam persoalan ini, karena Q adalah positip (semi)definit, maka penyelesaiannya dijamin merupakan penyelesaian global (optimum). Catatan bahwa jika suatu penyelesaian layak dapat diperoleh untuk persamaan (3.12) dan (3.13), dengan menjaga hubungan dari persamaan (3.14) dan (3.15) maka persoalan terpecahkan.

Suatu metode sederhana yang efisien untuk memecahkan persyaratan Kuhn-Tucker (3.12) sampai (3.15) telah dikembangkan dan dikenal sebagai metode complementary pivot. Metode ini secara komputasional lebih menarik dari pada metode-metode lain jika Q adalah positip semidefinit. Metode ini dikembangkan sebagai metode umum untuk memecahkan jenis persoalan khusus yang dikenal sebagai persoalan complementary yangmana akan dibahas pada seksi berikut.

III.3. Persoalan complementary

Anggaplah persoalan umum untuk mendapatkan penyelesaian nonnegatip dari suatu sistim persamaan sebagai berikut :

Dapatkan vektor w dan z sedemikian sehingga

$$w = Mz + q \tag{3.16}$$

$$w \geq 0, \quad z \geq 0 \tag{3.17}$$

$$w^T z = 0 \tag{3.18}$$

Dimana M adalah matrik kuadrat (n x n) dan w, z, q adalah vektor kolom berdimensi n.

Persoalan di atas dikenal sebagai persoalan complementary. Catatan bahwa tidak ada fungsi obyektif untuk memaksimumkan atau meminimumkan di dalam perumusan ini. Kondisi (3.16), menyatakan suatu sistim persamaan linier simultan; kondisi (3.17) mengharuskan penyelesaian terhadap kondisi (3.16) adalah nonnegatip; kondisi (3.18) menyatakan secara tidak langsung $w_i z_i = 0$ untuk semua $i = 1, 2, \ldots, n$ karena $w_i, z_i \geq 0$. Jadi terdapat sebuah constraint nonlinier tunggal.

Salah satu aplikasi dari persoalan complementary adalah untuk memecahkan persoalan program kuadratik dengan mengkonversikannya ke dalam persoalan complementary yang ekuivalen. Anggaplah persoalan program kuadratik berbentuk :

Minimumkan : $f(x) = cx + c^T Q x$

Dengan constraints : $Ax \geq b$

$x \geq 0$

dimana Q adalah matrik (n x n). Diasumsikan Q adalah simetri, dan positip definit atau positip semidefinit.

Di dalam notasi matrik, kondisi optimalitas Kuhn-Tucker untuk program kuadratik konvek dapat ditulis sebagai berikut :

Dapatkan vektor x, μ, v, s sedemikian sehingga

$$v = 2Qx - A^T\mu + c \tag{3.19}$$

$$s = -Ax + b \tag{3.20}$$

$$x, \mu, v, s \geq 0$$

$$v^T x + s^T \mu = 0$$

Dengan membandingkan sistim persamaan di atas terhadap persoalan complementary, dapat dilihat bahwa

$$w = \begin{bmatrix} v \\ s \end{bmatrix}, \quad z = \begin{bmatrix} x \\ \mu \end{bmatrix}, \quad M = \begin{bmatrix} 2Q & -A^T \\ A & 0 \end{bmatrix}, \quad q = \begin{bmatrix} c \\ -b \end{bmatrix}$$

Jadi suatu penyelesaian optimal untuk program kuadratik konvek dapat diperoleh dengan memecahkan persoalan complementary yang bersesuaian sebagaimana ditunjukkan di atas. Perlu dicatat bahwa matrik M adalah positip semidefinit, karena Q adalah positip definit atau positip semidefinit.

III.4. Algoritma complementary pivot

Anggaplah suatu persoalan complementary sebagai berikut :

Dapatkan vektor w dan z sedemikian sehingga

$$w = Mz + q$$

$$w, z \geq 0$$

$$w^T z = 0$$

Definisi

- *Penyelesaian layak*. Suatu penyelesaian nonnepatip (w, z) terhadap sistim persamaan $w = Mz + q$ dinamakan penyelesaian layak terhadap persoalan complementary.
- *Penyelesaian complementary*. Suatu penyelesaian layak (w, z) terhadap persoalan complementary yangmana juga memenuhi kondisi complementary $w^T z = 0$ dinamakan penyelesaian complementary.

Kondisi $w^T z = 0$ adalah ekuivalen dengan $w_i z_i = 0$ untuk semua i. Variabel w_i dan z_i untuk setiap i dinamakan pasangan complementary dari variabel-variabel. Catatan bahwa jika elemen dari vektor q adalah nonnegatip, maka terdapat sebuah penyelesaian complementary yang diberikan oleh $w = q$, $z = 0$. Sehingga persoalan complementary tidak sederhana jika sekurang-kurangnya satu elemen q adalah negatip. Ini berarti bahwa penyelesaian basic awal diberikan oleh $w = q$, $z = 0$ tidak layak terhadap persoalan complementay walaupun memenuhi kondisi complementary $w^T z = 0$

Saat ini tidak ada algoritma penyelesaian umum untuk memecahkan persoalan complementary. Jika matrik M memenuhi sifat-sifat khusus tertentu, suatu metode complementary pivot telah dikembangkan oleh Lemke untuk menentukan penyelesaian complementary. Secara khusus, metode complementary pivot dijamin memperoleh penyelesaian complementary jika M memenuhi salah satu dari sifat-sifat berikut :

1. Semua elemen M adalah positip.
2. Matrik M adalah positip definit.
3. Semua determinan utama dari matrik M adalah positip.

Yang dimaksud dengan determinan utama dari suatu matrik adalah determinan dari minor utama matrik tersebut. Sedangkan pengertian minor utama dari suatu matrik ($n \times n$) adalah submatrik berukuran ($k \times k$) yang didapat dengan menghilangkan ($n - k$) baris dan kolom terakhir dari matrik tersebut.

Sebelumnya telah dinyatakan bahwa persoalan pemrograman kuadratik bisa diubah menjadi persoalan complementary dimana matrik M adalah positip semidefinit. Di dalam masalah ini metode complementary pivot dijamin berakhir dengan sebuah penyelesaian complementary hanya jika terdapat sebuah penyelesaian untuk persoalan khusus tersebut. Dengan kata lain, adalah mungkin untuk persoalan complementary tidak memiliki penyelesaian karena beberapa program kuadratik mungkin tidak memiliki penyelesaian optimal.

Metode complementary pivot dimulai dengan penyelesaian dasar tak layak yang diberikan oleh $w = q$, $z = 0$. Untuk menjadikan penyelesaian nonnegatip, sebuah variabel artificial z_0 ditambahkan pada sebuah nilai yang cukup positip pada masing-masing persamaan di dalam sistim $w - Mz + q$, sehingga konstanta pada sisi ruas kanan ($q_i + z_0$) menjadi nonnegatip. Nilai z_0 akan sama dengan harga mutlak dari q_i yang paling negatip. Sekarang diperoleh sebuah penyelesaian dasar yang diberikan oleh :

$$w_i = q_i + z_0 \ , \quad z_i = 0 \ \text{untuk semua} \ i = 1, 2, \ldots, n$$

$$z_0 = - \min_i (q_i)$$

Walaupun penyelesaian ini adalah nonnegatip, memenuhi semua constraints dan persyaratan complementary ($w_i z_i = 0$), penyelesaian tersebut tidak

layak karena terdapat variabel artificial yang bernilai positip. Penyelesaian yang demikian dinamakan almost complementary solution.

Tahap pertama di dalam metode complementary pivot adalah mencari almost complementary solution, dengan merubah sistim persamaan semula ($w = Mz + q$) dengan sebuah variabel artificial z_0 sebagai berikut :

$$w - Mz - ez_0 = q$$

$$w, z, z_0 \geq 0$$

$$w^T z = 0$$

dimana

$$\underset{(n \times 1)}{e} = (1, 1, \ldots, 1)^T$$

atau bisa juga dinyatakan dengan

$$w - A\bar{z} = q$$

$$w^T \bar{z} = 0 \quad ; \quad w, \bar{z} \geq 0$$

dimana

$$A = (e, M) \quad ; \quad \bar{z}^T = (z_0, z)$$

Jadi, tabel awal untuk persoalan di atas adalah sebagai berikut :

Tabel A

Basis	w_1 . . . w_s . . . w_n	z_1 . . . z_s . . . z_n . . . z_0	q
w_1	1 0 0	$-m_{11}$ $-m_{1s}$ $-m_{1n}$ -1	q_1
w_s	0 1 0	$-m_{s1}$ $-m_{ss}$ $-m_{sn}$ -1	q_s
w_n	0 0 1	$-m_{n1}$ $-m_{ns}$ $-m_{nn}$ -1	q_n

dimana m_{ij} adalah elemen dari matrik M.

- **Tahap I.**

Untuk menentukan penyelesaian almost complemenrary awal, variabel z_0 masuk ke dalam basis menggantikan variabel basic dengan nilai paling negatip. Misalkan $q_s = \min q_i < 0$, berarti z_0 menggantikan w_s dari basis. Dengan melakukan operasi pivot dihasilkan tabel berikut :

Tabel B

Basis	w_1 . . . w_s . . . w_n	z_1 . . . z_s . . . z_n . . . z_0	q
w_1	1 -1 0	$-m'_{11}$ $-m'_{1s}$ $-m'_{1n}$ -1	q'_1
z_0	0 -1 0	$-m'_{s1}$ $-m'_{ss}$ $-m'_{sn}$ -1	q'_s
w_n	0 -1 1	$-m'_{n1}$ $-m'_{ns}$ $-m'_{nn}$ -1	q'_n

dimana

$$q'_s = -q_s \; ; \; q'_i = q_i - q_s \quad \text{untuk semua } i \neq s \; ;$$

$$m'_{sj} = m_{sj} \quad \text{untuk semua } j = 1, \ldots, n \; ;$$

$$m'_{ij} = m_{ij} + m_{sj} \quad \text{untuk semua } j = 1, \ldots, n \text{ dan } i \neq s \; .$$

Catatan :

1. $q_i \geq 0$, $i = 1, \ldots, n$
2. Penyelesaian dasar $w_1 = q'_1, \ldots, w_{s-1} = q'_{s-1}, z_0 = q'_s, w_{s+1} = q'_{s+1}$, . . . , $w_n = q'_n$ dan semua variabel lainnya yang bernilai awal nol adalah penyelesaian almost complementary.
3. Penyelesaian almost complementary berubah menjadi penyelesaian complementary apabila z_0 bernilai nol.

Pada dasarnya, algoritma complementary pivot melakukan proses untuk mendapatkan penyelesaian almost complementary sampai z_0 menjadi nol. Untuk melakukan ini, perubahan basis harus dilakukan sedemikian

sehingga kondisi berikut terpenuhi :

A. Complementary antara variabel-variabel harus dipertahankan, yaitu $w_i z_i = 0$ untuk $i = 1, \ldots, n$.

B. Penyelesaian dasar tetap non negatip, yaitu konstanta sisi ruas kanan di dalam tabel harus nonnegatip.

- **Tahap II.**

Untuk memenuhi kondisi A, variabel w_s dan z_s keduanya keluar dari basis di dalam tabel 1. Selama salah satu dari variabel tersebut menjadi variabel basic, complementary antara w dan z akan tetap terjaga. Karena w_s keluar dari basis, maka z_s dipilih untuk masuk ke dalam basis. Jadi, terdapat aturan yang sederhana untuk memilih variabel nonbasic yang akan masuk ke dalam basis pada tabel berikutnya, yaitu complement dari variabel basic yang telah keluar dari basis pada tabel semula. Ini disebut aturan complementary.

Setelah memilih variabel yang masuk ke dalam basis, selanjutnya dipilih variabel basic yang akan keluar. Agar kondisi B terpenuhi, maka penentuan variabel yang akan meninggalkan basis dilakukan dengan pengujian rasio minimum sebagai berikut :

$$\frac{q'_i}{m'_{is}} \quad \text{untuk semua } i = 1, \ldots, n \text{ dimana } m'_{is} > 0$$

Misalkan

$$\frac{q'_k}{m'_{ks}} = \underset{m'_{is} > 0}{\text{Minimum}} \left(\frac{q'_i}{m'_{is}}\right)$$

Ini berarti bahwa variabel basic w_k meninggalkan basis dan diganti oleh z_s. Sekarang didapatkan tabel baru dengan melakukan operasi pivot dengan m'_{ks} sebagai elemen pivot. Operasi pivot pada elemen pivot a_{ks} dilakukan sebagai berikut :

$$a'_{ij} = a_{ij} - \frac{a_{kj} x\, a_{is}}{a_{ks}}$$

$$q'_i = q_i - \frac{a_{kj} x\, q_k}{a_{ks}}$$

- **Tahap III.**

Karena w_k meninggalkan basis, variabel z_k dibawa menuju basis dengan aturan complementary, dan perubahan basis dilakukan seperti sebelumnya sampai terjadi satu dari dua hal yang menunjukkan selesainya algoritma :

- Rasio minimum diperoleh pada kolom s dan z_0 meninggalkan basis. Penyelesaian dasar yang dihasilkan setelah melakukan operasi pivot adalah penyelesaian complementary.
- Pengujian rasio minimum gagal, karena semua komponen di dalam kolom pivot adalah nonpositip. Ini berarti bahwa tidak terdapat penyelesaian bagi persoalan complementary. Dalam kasus ini, dikatakan bahwa persoalan complementary menghasilkan ray solution.

Sebagai ilustrasi tentang penggunaan metode complementary pivot di dalam memecahkan persoalan pemrograman kuadratik konvek diberikan sebuah contoh :

Minimumkan : $f(x) = -6x_1 + 2x_1^2 - 2x_1x_2 + x_2^2$

Dengan constraints : $-x_1 - x_2 \geq -2$

$$x_1 , x_2 \geq 0$$

Persoalan complementary yang ekuivalen dengan persoalan tersebut adalah:

$$M = \underset{(3\times3)}{\begin{pmatrix} 4 & -2 & 1 \\ -2 & 4 & 1 \\ -1 & 1 & 0 \end{pmatrix}} \qquad \text{dan} \qquad q = \begin{pmatrix} -6 \\ 0 \\ 2 \end{pmatrix}$$

Karena semua elemen q tidak nonnegatip, maka pada setiap persamaan ditambahkan variabel artificial z_0.

Tabel 1

Basis	w_1	w_2	w_3	z_1	z_2	z_3	z_0	q
w_1	1	0	0	-4	2	-1	-1	-6
w_2	0	1	0	2	-4	-1	-1	0
w_3	0	0	1	1	1	0	-1	2

Penyelesaian dasar awal adalah w_1=-6, w_2=0, w_3=2, $z_1 = z_2 = z_3 = z_0$=0. Penyelesaian almost complementary didapatkan dengan cara mengganti w_1 dengan z_0 seperti ditunjukkan pada tabel 2. Penyelesaian almost complementary diberikan oleh z_0=6, w_2=6, w_3=8, $z_1 = z_2 = z_3 = w_1 = 0$.

tabel 2

Basis	w_1	w_2	w_3	z_1	z_2	z_3	z_0	q
z_0	-1	0	0	4	-2	1	1	6
w_2	-1	1	0	**6**	-6	0	0	6
w_3	-1	0	1	5	-1	1	0	8

Karena pasangan complementary (w_1, z_1) keluar dari basis, baik w_1 maupun z_1 bisa dijadikan variabel basic tanpa mempengaruhi complementarity dari semua pasangan variabel ($w_i z_i = 0$). Karena w_1 meninggalkan basis, maka z_1 masuk ke dalam basis. Dengan menggunakan pengujian rasio minimum, diperoleh rasio (6/4, 6/6, 8/5). Ini berarti bahwa z_1 menggantikan w_2 di dalam basis. Tabel 3 memberikan penyelesaian almost complementary yang baru setelah dilakukan operasi pivot.

tabel 3

Basis	w_1	w_2	w_3	z_1	z_2	z_3	z_0	q
z_0	$-\frac{1}{3}$	$-\frac{2}{3}$	0	0	2	1	1	2
z_1	$-\frac{1}{6}$	$\frac{1}{6}$	0	1	-1	0	0	1
w_3	$-\frac{1}{6}$	$-\frac{5}{6}$	1	0	4	1	0	3

Dengan menggunakan aturan complementary, z_2 dipilih sebagai variabel basic berikutnya (w_2 telah meninggalkan basis). Pengujian rasio minimum menentukan bahwa w_3 harus meninggalkan basis. Penyelesaian almost complementary berikutnya setelah operasi pivot ditunjukkan di dalam tabel 4.

tabel 4

Basis	w_1	w_2	w_3	z_1	z_2	z_3	z_0	q
z_0	$-\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{1}{2}$	0	0	$\frac{1}{2}$	1	$\frac{1}{2}$
z_1	$-\frac{5}{24}$	$-\frac{1}{24}$	$\frac{1}{4}$	1	0	$\frac{1}{4}$	0	$\frac{7}{4}$
z_2	$-\frac{1}{24}$	$-\frac{5}{24}$	$\frac{1}{4}$	0	1	$\frac{1}{4}$	0	$\frac{3}{4}$

Sesuai dengan aturan complementary, z_3 menjadi variabel basic berikutnya menggantikan w_3. Penerapan uji rasio minimum mengharuskan penggantian z_0 dari basis. Karena z_0 meninggalkan basis, maka iterasi berakhir dan diperoleh penyelesaian complementary seperti ditunjukkan di dalam tabel 5.

tabel 5

Basis	w_1	w_2	w_3	z_1	z_2	z_3	z_0	q
z_3	$-\frac{1}{2}$	$-\frac{1}{2}$	-1	0	0	1	2	1
z_1	$-\frac{1}{12}$	$-\frac{1}{12}$	$\frac{1}{2}$	1	0	0	$-\frac{1}{2}$	$\frac{3}{2}$
z_2	$-\frac{1}{12}$	$-\frac{1}{12}$	$\frac{1}{2}$	0	1	0	$-\frac{1}{2}$	$\frac{1}{2}$

Penyelesaian complementary diberikan oleh $z_1=3/2$, $z_2=1/2$, $z_3=1$, $w_1=w_2=w_3=0$. Sehingga penyelesaian optimal dari program kuadratik menjadi :

$$\dot{x}_1 = \frac{3}{2} \qquad \dot{x}_2 = \frac{1}{2} \qquad \text{dan} \qquad f(\dot{x}) = -\frac{11}{2}$$

Diambil dari :

Philips Don T., Ravindran A., "Operation Research : Principles and Practice", John Wiley and Sons, NewYork.

BAB IV

PERUMUSAN PERMASALAHAN KE DALAM PROGRAM KUADRATIK

IV.1. Prosedur perhitungan numerik tahanan gelombang

Formula Integral Michell untuk tahanan gelombang merupakan fungsi kuadrat dari offsets lambung kapal *y*. Pada bagian ini akan diuraikan langkah-langkah perumusan formula tersebut ke dalam bentuk yang lebih sederhana, yaitu bentuk kuadratik standar [5]. Dengan bentuk yang lebih sederhana tersebut, disamping akan memudahkan di dalam menganalisa persoalan optimisasi yang ada juga akan mempermudah di dalam perhitungan numerik dengan komputer.

Seperti dijelaskan pada bab II.4 formula integral Michell untuk tahanan gelombang dari sebuah kapal pada kecepatan konstan *V* dan sarat *T* dinyatakan dalam persamaan :

$$R_W = \frac{4\rho g^2}{\pi V^2} \int_1^\infty \frac{\lambda^2}{(\lambda^2 - 1)^{1/2}} \left[I^2(\lambda) + J^2(\lambda) \right] d\lambda \qquad (4.1)$$

dimana

$$\begin{matrix} I(\lambda) \\ J(\lambda) \end{matrix} = \int_0^T \exp\left[\frac{g}{V^2} \lambda^2 (\zeta - T) \right] \cdot \left[\int_0^L H_\xi(\xi, \zeta) \begin{matrix} \cos \\ \sin \end{matrix} \left(\frac{g}{V^2} \lambda \xi \right) d\xi \right] d\zeta$$

Misalkan L, $B = 2b$, dan T berturut-turut adalah panjang, lebar, dan sarat kapal. Selanjutnya dipergunakan variabel-variabel nondimensional seperti berikut :

$$x = \xi/L \ , \ y = \eta/b \ , \ z = \zeta/T$$

Misalkan $h(x, z) = (1/b) H(\xi, \zeta)$ adalah hull function, maka slope function adalah

$$h_x(x, z) = \frac{L}{b} II(\xi, \zeta)$$

Untuk bilangan Froude $F_n = \dfrac{V}{\sqrt{gL}}$ didefinisikan bilangan gelombang nondimensional

$$\gamma_0 = \frac{g L}{2V^2} = \tfrac{1}{2} F_n{}^2$$

Maka koefisien tahanan gelombang nondimensional untuk bilangan Froude F_n dan perbandingan sarat-panjang T/L dapat ditulis sebagai

$$C_w = \frac{R_w}{\left(\dfrac{8\rho g}{\pi} \dfrac{B^2 T^2}{L}\right)}$$

$$= \frac{\gamma_0}{4} \int_1^\infty \frac{\lambda^2}{(\lambda^2 - 1)^{1/2}} \left[P^2(\lambda) + Q^2(\lambda)\right] d\lambda \qquad (4.2)$$

dimana

$$\begin{matrix} P(\lambda) \\ Q(\lambda) \end{matrix} = \int_0^1 \exp\left[-2\lambda^2 \gamma_0 \left(\frac{T}{L}\right)(1 - z)\right] \cdot \left[\int_0^1 h_x(x, z) \begin{matrix} \cos \\ \sin \end{matrix} (2\lambda\gamma_0 x)\, dx\right] dz$$

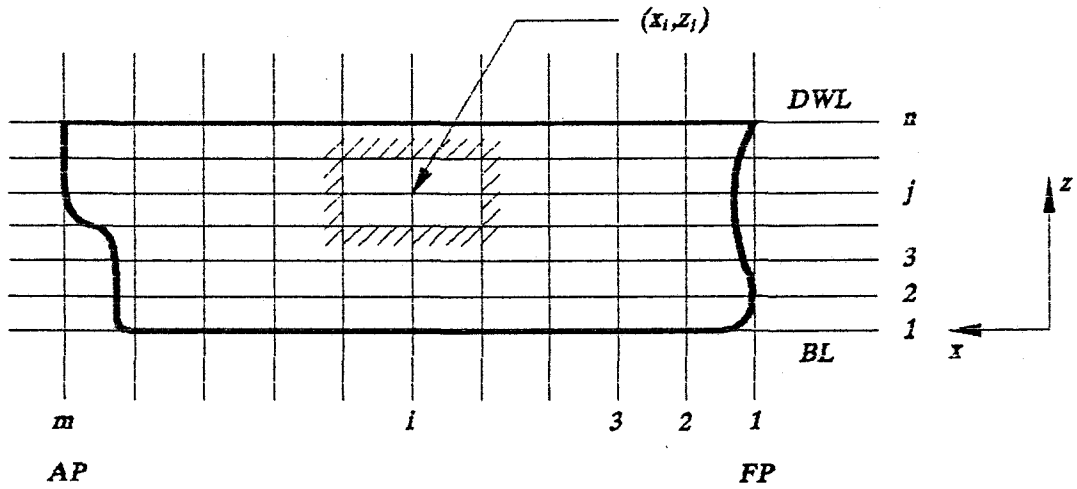
Setelah mengeliminasi singularity pada $\lambda = 1$ dengan memisalkan $\lambda = u^2 + 1$, persamaan (4.2) menjadi

$$C_w = \frac{\gamma_0}{2} \int_0^{\infty} \frac{(u^2+1)^2}{(u^2+2)^{1/2}} \left[\tilde{P}^2(u) + \tilde{Q}^2(u) \right] du \qquad (4.3)$$

dimana

$$\begin{matrix} \tilde{P}(u) \\ \tilde{Q}(u) \end{matrix} = \int_0^1 \exp\left[-2\gamma_0 \left(\frac{z}{L} \right) (1-z)(u^2+1)^2 \right] \cdot \left[\int_0^1 h_x(x,z) \begin{matrix} \cos \\ \sin \end{matrix} \left(2(u^2+1)\gamma_0 x \right) dx \right] dz$$

Berikutnya akan diperkenalkan suatu fungsi yang disebut tent function [1]. Sebuah kapal biasanya didefinisikan dengan menggunakan sejumlah station dan garis air, seperti grid frame work pada gambar (4.1). Jarak antara station atau garis air tidak perlu sama. Sebagai perjanjian, station pertama adalah titik terdepan sedangkan station terakhir adalah titik paling belakang. Garis air pertama adalah garis dasar sedangkan garis air



Station : $i = 1, 2, \ldots, m$; Garis air : $j = 1, 2, \ldots, n$.

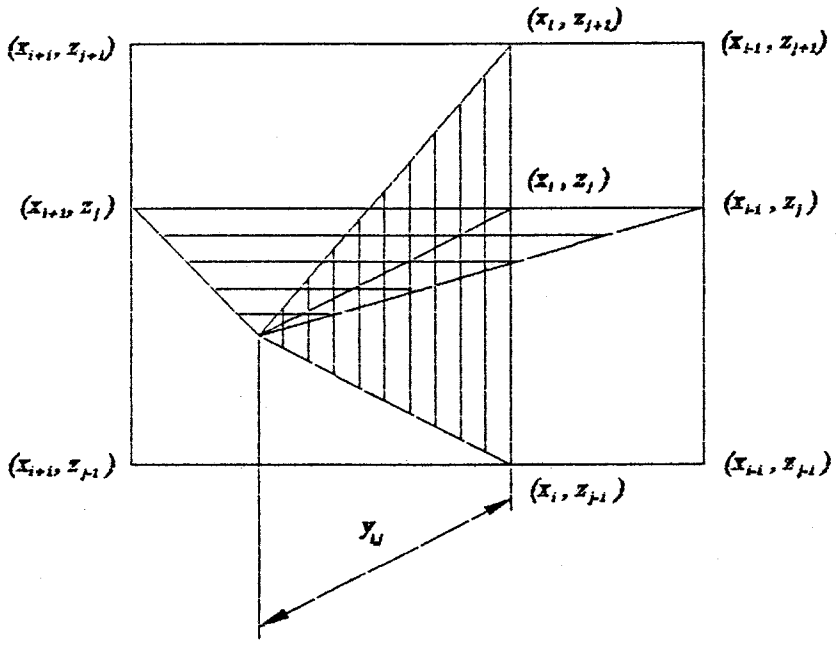
Gambar 4.1. Grid framework

terakhir adalah garis air perencanaan (DWL). Pada elemen empat persegi panjang yang berpusat pada (x_i, z_j), dan dibatasi oleh station $(i-1)$, $(i+1)$ serta garis air $(j-1)$, $(j+1)$ akan didefinisikan sebuah unit tent function yang bernilai satu pada (x_i, z_j) dan bernilai nol pada batas elemen.

Unit tent function untuk (x_i, z_j) didefinisikan sebagai berikut :

$$h^{(i,j)}(x,z) \equiv \begin{cases} \left(1-\dfrac{x_i-x}{x_i-x_{i-1}}\right)\left(1-\dfrac{z_j-z}{z_j-z_{j-1}}\right) \\ x_{i-1} \le x \le x_i, z_{j-1} \le z \le z_j \\ \left(1-\dfrac{x_i-x}{x_i-x_{i-1}}\right)\left(1-\dfrac{z_j-z}{z_j-z_{j+1}}\right) \\ x_{i-1} \le x \le x_i, z_j \le z \le z_{j+1} \\ \left(1-\dfrac{x_i-x}{x_i-x_{i+1}}\right)\left(1-\dfrac{z_j-z}{z_j-z_{j-1}}\right) \\ x_i \le x \le x_{i+1}, z_{j-1} \le z \le z_j \\ \left(1-\dfrac{x_i-x}{x_i-x_{i+1}}\right)\left(1-\dfrac{z_j-z}{z_j-z_{j+1}}\right) \\ x_i \le x \le x_{i+1}, z_j \le z \le z_{j+1} \\ 0, \text{untuk yang lain} \end{cases} \tag{4.4}$$

Catatan, walaupun $h^{(i,j)}(x, z)$ bukan fungsi linier, tetapi h linier tehadap x untuk z tetap dan linier terhadap z untuk x tetap di dalam setiap kuadran dari elemen empat persegi panjang tersebut. Gambaran tentang sebuah tent function dapat dilihat pada gambar (4.2).



Gambar. 4.2 Satu satuan tent function

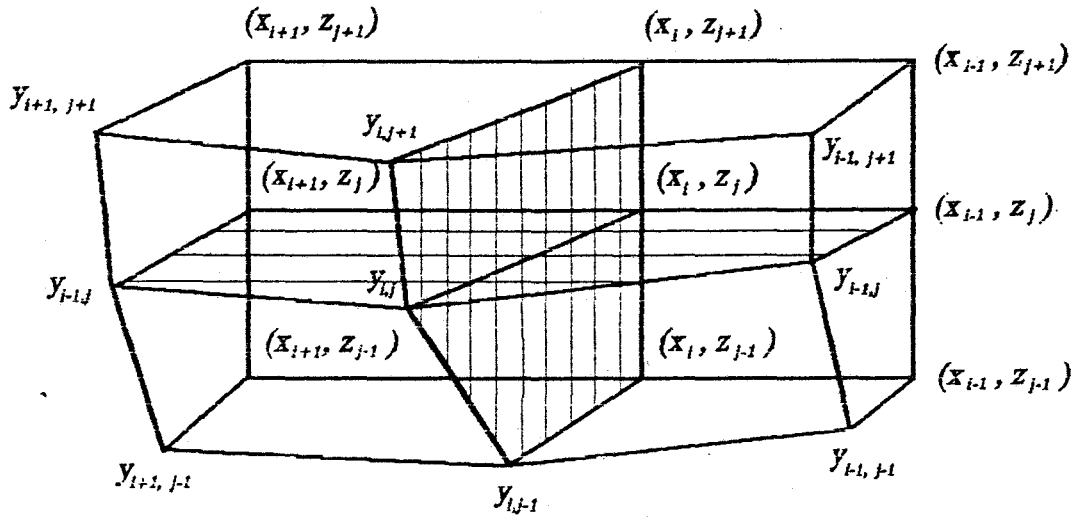
Selanjutnya bentuk tent function digunakan untuk membentuk suatu fungsi untuk pendekatan permukaan lambung $f(x, y)$. Jika y_{ij} adalah offset lambung pada (x_i, z_j), didefinisikan fungsi pendekatan

$$\hat{h}(x,z) = \sum_i \sum_j y_{ij} h^{(i,j)}(x,z) \qquad (4.4)$$

Ini mengikuti sifat dari tent function secara tepat bahwa pada grid point (x_i, z_j) adalah :

$$\hat{h}(x_i, z_j) = h(x_i, z_j) \qquad (4.6)$$

Antara (x_i, z_j) dan semua titik yang berdekatan dengan grid point (x_{i-1}, z_j), (x_i, z_{j+1}), (x_{i+1}, z_j), (x_i, z_{j-1}) pendekatannya adalah sebuah garis lurus. Sepanjang setiap garis air pendekatannya adalah linier terhadap x di antara station-station, dan sepanjang setiap garis station pendekatannya



Gambar 4.3 Bagian dari lambung yang dibentuk dari tent function

adalah linier terhadap z di antara garis air. Gambar (4.3) menunjukkan bentuk pendekatan tersebut di dalam sebuah elemen segi empat. Dengan pendekatan ini persamaan (4.3) dapat direduksi menjadi bentuk yang lebih sederhana.

Dengan mensubstitusikan (4.5), persamaan(4.2) akan dibawa ke dalam bentuk fungsi

$$C_i(\lambda;\gamma_0) = \frac{1}{x_i - x_{i+1}} \int_{x_i}^{x_{i+1}} \cos(2\lambda\gamma_0 x)\, dx + \frac{1}{x_i - x_{i-1}} \int_{x_{i-1}}^{x_i} \cos(2\lambda\gamma_0 x)\, dx$$

$$= -\frac{1}{2\gamma_0\lambda}\left[\frac{1}{x_{i+1} - x_i}(\sin 2\lambda\gamma_0 x_{i+1} - \sin 2\lambda\gamma_0 x_i)\right.$$

$$\left. -\frac{1}{x_i - x_{i-1}}(\sin 2\lambda\gamma_0 x_i - \sin 2\lambda\gamma_0 x_{i-1})\right] \qquad (4.7)$$

$$S_i(\lambda;\gamma_0) = \frac{1}{x_i - x_{i+1}} \int_{x_i}^{x_{i+1}} \sin(2\lambda\gamma_0 x)\,dx + \frac{1}{x_i - x_{i-1}} \int_{x_{j-1}}^{x_j} \sin(2\lambda\gamma_0 x)\,dx$$

$$= -\frac{1}{2\gamma_0\lambda}\left[\frac{1}{x_{i+1}-x_i}(\cos 2\lambda\gamma_0 x_{i+1} - \cos 2\lambda\gamma_0 x_i)\right.$$

$$\left. -\frac{1}{x_i - x_{i-1}}(\cos 2\lambda\gamma_0 x_i - \cos 2\lambda\gamma_0 x_{i-1})\right] \qquad (4.8)$$

$$E_j(\lambda;\gamma_0,T/L) = \int_{z_j}^{z_{j+1}} \exp\left[-2\lambda^2\gamma_0(T/L)(1-z)\right] \cdot \left[1 - \frac{z_j - z}{z_j - z_{j+1}}\right] dz$$

$$+ \int_{z_{j-1}}^{z_j} \exp\left[-2\lambda^2\gamma_0(T/L)(1-z)\right] \cdot \left[1 - \frac{z_j - z}{z_j - z_{j-1}}\right] dz$$

$$= \frac{1}{\left[2\lambda^2\gamma_0(T/L)\right]^2} \cdot \left\{\frac{1}{z_{j+1}-z_j}\left[\exp(-2\lambda^2\gamma_0(T/L)(1-z_{j+1}))\right.\right.$$

$$\left. -\exp(-2\lambda^2\gamma_0(T/L)(1-z_j))\right] - \frac{1}{z_j - z_{j-1}}\left[\exp(-2\lambda^2\gamma_0(T/L)(1-z_j))\right.$$

$$\left.\left. -\exp(-2\lambda^2\gamma_0(T/L)(1-z_{j-1}))\right]\right\} \qquad (4.9)$$

Kemudian dengan (4.2), (4.5), dan (4.6) didapatkan

$$P(\lambda) = \sum_i \sum_j y_{ij} \int dx\, h_x^{(i,j)}(x,z) \cdot \cos(-2\lambda\gamma_0 x)$$

$$\int dz\, \exp\left[2\lambda\gamma_0(T/L)(1-z)\right]$$

$$= \sum_{ij} y_{ij} \int_{x_i}^{x_{i+1}} \left\{\frac{1}{x_i - x_{i+1}} \cdot \cos(2\lambda\gamma_0 x)\,dx \int_{x_j}^{x_j} \frac{1}{x_i - x_{j-1}} \cdot \cos(2\lambda\gamma_0 x)\,dx\right\}$$

$$\cdot \left\{\int_{z_j}^{z_{j+1}} \exp\left[-2\lambda^2\gamma_0(T/L)(1-z)\right] \cdot \left[1 - \frac{z_j - z}{z_j - z_{j+1}}\right] dz\right.$$

$$\left. + \int_{z_{j-1}}^{z_j} \exp\left[-2\lambda^2\gamma_0(T/L)(1-z)\right] \cdot \left[1 - \frac{z_j - z}{z_j - z_{j-1}}\right] dz\right\}$$

$$= \sum_{ij} y_{ij} \cdot C_i(\lambda;\gamma_0) \cdot E_j(\lambda;\gamma_0,T/L)$$

Dengan cara yang sama didapatkan pula

$$Q(\lambda) = \sum_{ij} y_{ij} . S_i(\lambda ; \gamma_0) . E_j(\lambda ; \gamma_0, T/L)$$

Selanjutnya,

$$P^2(\lambda) = \sum_{ij} \sum_{kl} y_{ij} y_{kl} . C_i C_k E_j E_l$$

$$Q^2(\lambda) = \sum_{ij} \sum_{kl} y_{ij} y_{kl} . S_i S_k E_j E_l \qquad (4.10)$$

Dengan mensubstitusikan (4.10) ke dalam (4.3) didapatkan

$$C_w = \sum_{ij} \sum_{kl} y_{ij} y_{kl} . \left[\frac{\gamma_0}{2} \int_0^\infty \frac{(u^2+1)^2}{(u^2+2)^{1/2}} \left[(C_i C_k + S_i S_k) E_j E_l \right] du \right] \qquad (4.11)$$

Selanjutnya didefinisikan suatu fungsi tahanan universal

$$d_{ijkl}(\gamma_0, T/L) = \frac{\gamma_0}{2} \int_0^\infty \frac{(u^2+1)^2}{(u^2+2)^{1/2}} \left[(C_i C_k + S_i S_k) E_j E_l \right] du \qquad (4.12)$$

yang mana tidak tergantung pada offsets kapal tetapi hanya tergantung pada bilangan Froude dan perbandingan sarat-lebar.

Di dalam (4.11) dan (4.12) penjumlahan yang meliputi *i* dan *k* serta meliputi *j* dan *l* mempunyai batas yang sama. Dimensi penjumlahan tersebut dapat dikecilkan dua kali dengan menuliskan (4.11) ke dalam bentuk kuadratik berikut :

$$C_w = \sum_{m=1}^{n} \sum_{n=1}^{n} d_{mn} y_m y_n \qquad (4.13)$$

dimana offset lambung y_{ij} sama dengan y_m dengan

$$m = i + (j - 1) \text{ x (batas penjumlahan } i \text{)}$$

dan d_{ijkl} sama dengan d_{mn} dengan m seperti di atas dan

$$n = k + (l\text{-}1) \text{ x (batas penjumlahan } k)$$

Catatan bahwa

$$d_{mn} = d_{nm}$$

Batas penjumlahan dari m atau n adalah :

$$\tilde{n} = (\text{batas penjumlahan } i) \text{ x (batas penjumlahan } j)$$

Persamaan (4.13) dapat juga ditulis dalam bentuk matrik sebagai berikut :

$$C_W = y^T . \tilde{D} . y \qquad (4.14)$$

dimana

y = vektor kolom dari offsets

y^T= transpose dari vektor y

$\tilde{D}$= matrik tahanan yang simetris

Integral dari (4.12) dapat dihitung secara numerik dengan menggunakan aturan Simpson dengan memilih interval dan batas integral yang memadai.

IV.2. Perumusan persoalan program kuadratik

Untuk menyelesaikan persoalan optimisasi, fungsi obyektif $C_W(y)$ harus dirumuskan dalam bentuk standar. Pada pembahasan sebelumnya diuraikan penyederhanaan integral Michell menjadi bentuk kuadratik standar (4.14) dengan menggunakan tent function.

Karena pertimbangan atau keterikatan terhadap beberapa hal, tidak semua offsets lambung harus dioptimumkan. Pada beberapa bagian dari lambung, offsets sering kali dibuat tetap, misalnya daerah buritan yang banyak keterikatan terhadap ruang mesin dan propeller.

Misalkan $y_1, y_2, \ldots, y_p$ adalah offsets lambung yang tetap, sedangkan y_{p+1}, $y_{p+2}, \ldots, y_{\tilde{n}}$ adalah offsets yang akan dioptimumkan. Maka persamaan (4.14) berubah menjadi

$$C_w = \sum_{i=1}^{\tilde{n}} \sum_{j=1}^{p} d_{ij} y_i y_j + \sum_{i=1}^{\tilde{n}} \sum_{j=p+1}^{\tilde{n}} d_{ij} y_i y_j$$

$$= \sum_{i=1}^{\tilde{n}} y_i \left[\sum_{j=1}^{p} d_{ij} y_j \right] + \sum_{j=p+1}^{\tilde{n}} y_j \left[\sum_{i=1}^{\tilde{n}} d_{ij} y_i \right] \qquad (4.15)$$

Untuk variabel yang tetap didefinisikan konstanta

$$c_i = \sum_{j=1}^{p} d_{ij} y_j, \text{ untuk setiap } i \qquad (4.16)$$

Sehingga (4.15) menjadi

$$C_w = \sum_{i=1}^{p} c_i y_i + \sum_{i=p+1}^{\tilde{n}} c_i y_i + \sum_{j=p+1}^{\tilde{n}} c_j y_j + \sum_{j=p+1}^{\tilde{n}} y_j \left[\sum_{i=p+1}^{\tilde{n}} d_{ij} y_i \right] \qquad (4.17)$$

$$= C_{w0} + 2 \sum_{i=p+1}^{\tilde{n}} c_i y_i + \sum_{i=p+1}^{\tilde{n}} \sum_{j=p+1}^{\tilde{n}} d_{ij} y_i y_j$$

dimana $C_{w0} = a$ merupakan suatu bentuk konstanta.

Di dalam persoalan khusus, jika semua variabel dioptimumkan, maka $p = 0$, $c_i = 0$, $C_{w0} = 0$, sehingga persamaan (4.17) sama dengan (4.13).

Selanjutnya persamaan (4.17) diubah ke dalam bentuk kuadratik standar. Misalkan $r = i - p$, $s = j - p$, dan $n = \tilde{n} - p$, sehingga didapatkan bentuk kuadratik standar dengan n variabel yang tidak diketahui :

$$C_w = C_{w0} + \sum_{r=1}^{n} (2c_r y_r) + \tfrac{1}{2} \sum_{r=1}^{n} \sum_{s=1}^{n} (2d_{rs}) y_r y_s \qquad (418)$$

atau di dalam bentuk matrik :

$$C_w = C_{w0} + C^T.y + y^T.D.y \qquad (4.19)$$

dimana

y = vektor n kolom dari offset

C = vektor n kolom dari koefisien linier dengan elemen 2 c_r

D = matrik tahanan n x n (simetris) dengan elemen d_{rs}

Karena $y^T.D.y \geq 0$ untuk semua y maka matrik D adalah positip semidefinit [5]. Sehingga fungsi kuadratik $C_w(y)$ adalah konvek.

Dengan menggunakan constraints pertidaksamaan linier

$$A.y \geq b, \quad y \geq 0 \qquad (4.20)$$

dimana

y = vektor n kolom

A = matrik kendala m x n

b = vektor m kolom

persamaan (4.19) dapat dioptimumkan dengan merubahnya ke dalam persoalan program kuadratik konvek [5] seperti berikut :

minimumkan

$$c^T.y + y^T.D.y \qquad (4.21)$$

dengan constraints

$$A.y \geq b, \quad y \geq 0$$

Setelah dirumuskan ke dalam persoalan kuadratik standar, selanjutnya metode Complementary Pivot [5] dapat digunakan untuk mendapatkan offsets lambung optimum y yang menghasilkan tahanan gelombang minimum.

IV.3. Meminimumkan tahanan gelombang untuk beberapa kecepatan kapal

Untuk meminimumkan tahanan gelombang pada beberapa kecepatan kapal, dapat dituliskan persamaan berikut [1] :

$$C_w = \sum_i g_i C_{wi} \tag{4.22}$$

dimana g_i adalah weighting factor.

Dari persamaan (4.14) pada suatu bilangan Froude tertentu didapatkan

$$C_{wi} = y^T . \tilde{D} . y \tag{4.23}$$

Dengan mensubstitusikan (4.23) ke dalam (4.22), didapatkan

$$\begin{aligned} C_w &= \sum_i g_i (y^T . \tilde{D} . y) \\ &= y^T \left[\sum_i g_i \tilde{D}_i \right] y \\ &= y^T . \hat{D} . y \end{aligned} \tag{4.24}$$

dimana

$$\hat{D} = \sum_i g_i \tilde{D}$$

adalah matrik tahanan ekuivalen. Selanjutnya $\hat{D}$ bisa dihitung dengan menggunakan skema komputasional yang sama seperti digambarkan sebelumnya untuk meminimumkan $C_W(y)$.

Untuk menentukan weighting factor g_i, dimisalkan energi total kapal yang digunakan dalam satu pelayaran untuk mengatasi tahanan gelombang adalah E_W, sehingga

$$E_W = \sum_i E_i \tag{4.25}$$

dimana E_i adalah energi yang diperlukan untuk mengatasi tahanan gelombang pada kecepatanV_i. Jika kecepatan rata-rata kapal adalah $\overline{V}$ maka persamaan (4.25) dapat pula ditulis sebagai berikut

$$C_w \cdot \overline{V} \cdot t_T = \sum_i C_{w_i} \cdot V_i \cdot t_i \tag{4.26}$$

dimana t_T adalah waktu pelayaran total dsn t_i adalah interval waktu untuk kecepatan c_i. Kemudian (4.26) dapat pula ditulis sebagai berikut

$$C_w \cdot S_T = \sum_i C_{w_i} \cdot S_i \tag{4.27}$$

dimana $S_T = \overline{c}_i \cdot t_T$ adalah jarak total yang dilalui untuk satu pelayaran, dan$S_i = c_i \cdot t_i$ adalah jarak yang dilalui untuk kecepatan $c_i$dalam waktu t_i.

Dari persamaan (4.22) dan (4.27) didapatkan harga weighting factor untuk C_{W_i}

$$g_i = \frac{S_i}{S_T} \tag{4.28}$$

Untuk kondisi yang lebih realstis dapat digunakan analisa statistik untuk mendapatkan distribusi kecepatan atau distribusi jarak di dalam perhitungan weighting factor.

IV.3. Perumusan constraints untuk permasalahan

Constraints yang digunakan di dalam persoalan optimisasi ini adalah constraints bentuk geometris lambung kapal. Semua constraints tersebut adalah linier terhadap offsets lambung y . Constraints harus dirumuskan dalam bentuk standar menurut persamaan (4.20)

Hasil perhitungan dipengaruhi oleh constraints yang digunakan. Suatu penyelesaian yang secara fisik mustahil bisa terjadi di dalam persoalan optimisasi. Bisa jadi perhitungan menghasilkan bentuk lambung yang bergelombang secara tak teratur. Agar didapatkan hasil yang lebih praktis, permasalahan optimisasi dari integral Michell harus menggunakan constraints yang tidak hanya koefisien-koefisien bentuk kapal, tetapi juga kondisi tambahan dari constraints pertidaksamaan berikut [1] :

$$0 \leq H(\xi,\zeta) \leq B \tag{4.29}$$

$$C \leq \frac{\partial H(\xi,\zeta)}{\partial \xi} \leq D \tag{4.30}$$

dimana

$H(\xi,\zeta)$ = hull function

B, C, D = konstanta

Misalkan sistim koordinat kapal seperti ditunjukkan pada gambar (2.5), dan offsets kapal merupakan besaran nondimensi sebagaimana dijelaskan di muka. Misalkan L, B, T adalah ukuran utama kapal yaitu berturut-turut panjang, lebar, dan sarat kapal. Kapal yang akan dioptimalkan untuk mendapatkan tahanan gelombang terkecil akan menggunakan salah satu atau kombinasi dari beberapa constraints geometrik berikut :

- **Batas atas offsets**

Harga offset yang tertentu $\tilde{y}_{ij}$ menjadi batas atas dari lambung kapal :

$$y_{ij} \leq \tilde{y}_{ij}, \quad \text{atau} \quad -y_{ij} \geq -\tilde{y}_{ij} \tag{4.31}$$

- **Batas bawah offsets**

Harga offset yang tertentu $\tilde{y}_{ij}$ menjadi batas bawah dari lambung kapal :

$$y_{ij} \geq \tilde{y}_{ij} \tag{4.32}$$

Untuk mengurangi jumlah constraints sehingga mempercepat proses perhitungan, akan lebih menguntungkan apabila di dalam persamaan (4.32) disubstitusikan persamaan berikut :

$$y_{ij} = \hat{y}_{ij} + \tilde{y}_{ij}, \quad \hat{y}_{ij} \geq 0 \tag{4.33}$$

Fungsi obyektif (4.21) selanjutnya berubah menjadi :

$$\begin{aligned} f(y) &= c^T.(\hat{y}+\tilde{y}) + (\hat{y}+\tilde{y})^T.D.(\hat{y}+\tilde{y}) \\ &= (c^T.\tilde{y} + c^T.\hat{y}) + (\tilde{y}^T.D.\tilde{y} + \tilde{y}^T.D.\hat{y} + \hat{y}^T.D.\hat{y}) \\ &= (c^T.\tilde{y} + \tilde{y}^T.D.\tilde{y}) + (c^T.\hat{y} + \tilde{y}^T.D.\hat{y}) + \hat{y}^T.D.\hat{y} \end{aligned} \tag{4.34}$$

Bagian pertama merupakan bentuk konstan yang bisa diabaikan di dalam persoalan program kuadratik. Sedangkan bagian kedua dan ketiga berturut-turut adalah bentuk linier dan bentuk kuadratik dari variabel baru $\hat{y}$. Setelah menghilangkan bentuk konstan dari persamaan di atas, didapatkan persoalan program kuadratik yang baru :

minimumkan :

$$\hat{c}^T.\hat{y} + \hat{y}^T.D.\hat{y}$$

dengan constraints : (4.35)

$$A.\hat{y} \geq \hat{b}, \quad \hat{y} \geq 0$$

dimana

$$\hat{c}^T = c^T + \tilde{y}^T.D$$

$$\hat{b} = b - A.\tilde{y}$$

- **Sudut garis air**

Sudut garis air, yaitu sudut yang di bentuk oleh tangent garis air dan center line, adalah lebih kecil atau sama dengan sudut batas atas $\tilde{\theta}$:

$$\frac{B}{2L}\left(\frac{y_{i+1,j}-y_{i,j}}{x_{i+1}-x_i}\right) \leq \tan\tilde{\theta}$$

atau

$$(y_{i+1,j}-y_{i,j}) \geq (x_{i+1}-x_i)\frac{2L}{B}\tan\tilde{\theta} \tag{4.36}$$

- **Sudut seksi**

Sudut seksi φ , yaitu sudut yang dibentuk oleh tangent garis station dengan garis dasar, adalah lebih besar atau sama dengan sudut batas bawah $\tilde{\varphi}$:

$$\frac{B}{2T}\left(\frac{y_{i+1,j}-y_{i,j}}{z_{j+1}-z_j}\right) \leq \cot\tilde{\varphi}$$

atau

$$(y_{i+1,j}-y_{i,j}) \geq (z_{j+1}-z_j)\frac{2T}{B}\cot\tilde{\varphi} \tag{4.37}$$

- **Koefisien bidang garis air**

Koefisien bidang garis air pada garis air *j* adalah :

$$C_{WP_j} = \frac{A_j}{BL} = \tfrac{1}{2}\sum_{i=1}^{m-1}(x_{i+1}-x_i)(y_{i+1,j}-y_{i,j})$$

atau

$$C_{WP_j} = \sum_{i=1}^{m} a_i . y_{i,j} \tag{4.38}$$

dimana

m = jumlah station (termasuk FP)

$a_i = \tfrac{1}{2}[x_{i+1}-x_{i-1}]$, $i = 2,\ldots,m\text{-}1$

$a_i = \tfrac{1}{2}[x_2-x_1]$, $i = 1$

$a_i = \tfrac{1}{2}[x_m-x_{m-1}]$, $i = m$

- **Koefisien midship**

Luas seksi pada station *i* adalah :

$$A_i = \frac{BT}{2} \sum_{j=1}^{n\ 1} (z_{j+1} - z_j)\,(y_{i,j+1} + y_{ij}) \tag{4.39}$$

dimana *n* adalah jumlah total garis air (termasuk garis dasar). Misalkan A_Φ adalah luas seksi pada midship (station Φ), maka koefisien midship dapat ditulis sebagai :

$$C_m = \frac{A_\Phi}{BT} = \tfrac{1}{2} \sum_{j=1}^{n-1} (z_{j+1} - z_j)\,(y_{\Phi,j+1} + y_{\Phi,j})$$

atau (4.40)

$$C_m = \sum_{j=1}^{n} b_j \cdot y_{\Phi,j}$$

dimana

$b_j = \tfrac{1}{2}\,[z_{j+1} - z_{j-1}]$, $j = 2, \ldots, n-1$

$b_j = \tfrac{1}{2}\,[z_2 - z_1]$, $j = 1$

$b_j = \tfrac{1}{2}\,[z_n - z_{n-1}]$, $j = n$

- **Koefisien block**

Volume displasemen dari kapal dapat diperoleh dengan menggunakan tent function. Volume dari elemen "tent" dengan offset y_{ij} (gambar 4.2) di dalam daerah $x_{i-1} \le x \le x_i$ dan $z_{j-1} \le z \le z_j$ adalah :

$$\text{Vol}_I = \int_{x_{i-1}}^{x_i} \int_{z_{j-1}}^{z_j} y_{ij} \left(1 - \frac{x_i - x}{x_i - x_{i-1}}\right) \left(1 - \frac{z_j - z}{z_j - z_{j-1}}\right) dz\ dx$$

$$= \frac{y_{ij}}{4} (x_i - x_{i-1})\,(z_j - z_{j-1}) \tag{4.41}$$

Dengan cara yang sama didapatkan :

$$\text{Vol II} = \frac{y_{ij}}{4}(x_i - x_{i-1})(z_{j+1} - z_j) \quad \text{untuk} x_{i-1} \le x \le x_i \,,\; z_j \le z \le z_{j+1}$$

$$\text{Vol III} = \frac{y_{ij}}{4}(x_{i+1} - x_i)(z_j - z_{j-1}) \quad \text{untuk} x_i \le x \le x_{i+1} \,,\; z_{j-1} \le z \le z_j$$

$$\text{Vol IV} = \frac{y_{ij}}{4}(x_{i+1} - x_i)(z_{j+1} - z_j) \quad \text{untuk} x_i \le x \le x_{i+1} \,,\; z_j \le z \le z_{j+1}$$

Dengan menggabungkan keempat elemen di atas didapatkan volume dari tent elemen *ij* dalam bentuk dimensional

$$\text{Vol}_{ij} = BLT \frac{y_{ij}}{2}(x_{i+1} - x_{i-1})(z_{j+1} - z_{j-1}) \tag{4.42}$$

maka volume total kapal adalah

$$V = BLT \sum_{i=1}^{m} a_i \sum_{j=1}^{n} b_j y_{ij} \tag{4.43}$$

dimana

$a_i = \frac{1}{2}[x_{i+1} - x_{i-1}]$, $i = 2, \ldots, m\text{-}1$

$a_i = \frac{1}{2}[x_2 - x_1]$, $i = 1$

$a_i = \frac{1}{2}[x_m - x_{m-1}]$, $i = m$

$b_j = \frac{1}{2}[z_{j+1} - z_{j-1}]$, $j = 2, \ldots, n\text{-}1$

$b_j = \frac{1}{2}[z_2 - z_1]$, $j = 1$

$b_j = \frac{1}{2}[z_n - z_{n-1}]$, $j = n$

Sehingga koefisien block, suatu bentuk nondimensional dari displasemen, dapat dinyatakan dengan

$$C_B = \frac{V}{BLT} = \sum_{ij} c_{ij} \cdot y_{ij} \tag{4.44}$$

dimana $c_{ij} = a_i\, b_j$, dengan a_i dan b_j seperti di atas

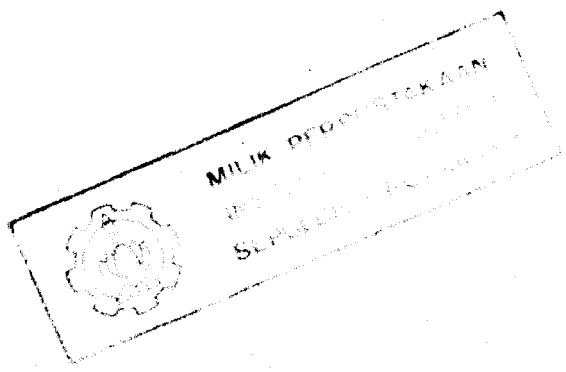
- **Koefisien prismatik**

Dari persamaan (4.40) dan (4.44) dapat juga diturunkan persamaan untuk koefisien prismatik kapal :

$$C_P = \frac{V}{A_\Phi L} = \frac{C_B}{C_m}$$

$$= \frac{\sum_{ij} c_{ij} \cdot y_{ij}}{\sum_{j} b_j \cdot y_{\Phi,j}} \qquad (4.45)$$

Beberapa constraint di atas dapat dipakai secara kombinasi menurut keperluan. Pemilihan constraints akan berpengaruh terhadap bentuk lambung optimal yang dihasilkan dan besar penurunan tahanan gelombang untuk bentuk tersebut, serta menentukan waktu yang diperlukan untuk perhitungan. Untuk itu penentuan constraints harus dilakukan dengan cermat. Jika tidak, akan dihasilkan penyelesaian yang tidak sesuai dengan keinginan, bahkan penyelesaian yang mustahil. Suatu persoalan tanpa penyelesaian bisa terjadi jika constraints yang digunakan tidak bersesuaian satu dengan yang lain. Jika hal ini terjadi, pemeriksaan ulang harus dilakukan terhadap formulasi constraints yang dipakai.



BAB V
ANALISA HASIL PERHITUNGAN

V.1. Validasi program

Sebelum bentuk lambung optimum dari hasil perhitungan di analisa, perlu dilakukan pengujian untuk mengetahui valid tidaknya program sehingga hasil perhitungan program dapat dipertanggungjawabkan. Ada tiga bagian dari program yang dianggap perlu untuk diuji, yaitu :

- Perhitungan tahanan gelombang.
- Perumusan program kuadratik.
- Penyelesaian program kuadratik.

□ Perhitungan tahanan gelombang

Untuk membuktikan bahwa perhitungan tahanan gelombang adalah valid, hasil perhitungan dari program (C_{W1}) dibandingkan dengan hasil perhitungan serupa di dalam referensi [1], (C_{W2}). Perhitungan dilakukan terhadap kapal series 60 block 60 dengan data T/L = 0.053. Hasil dari kedua perhitungan dapat dilihat pada tabel (5.1).

Adanya selisih dari kedua perhitungan dipengaruhi oleh tingkat ketelitian di dalam perhitungan, yaitu tergantung pada pemilihan interval dan batas integral yang digunakan di dalam perhitungan numerik matrik tahanan. Untuk perhitungan ini dipilih interval = 0.4 dan batas integral = 3.6, dengan pertimbangan bahwa hingga pada harga-harga tersebut hasil perhitungan mendekati konstan.

Dengan selisih perhitungan yang cukup kecil tersebut, dianggap keabsahan program bisa dipertanggung jawabkan.

Tabel 5.1

F_n	C_{w1}	C_{w2}	$\frac{(C_{w1}-C_{w2})}{C_{w2}} \cdot 100\%$
0.289	0.203	0.204	0.5
0.316	0.159	0.164	3.0

☐ Penyelesaian program kuadratik

Untuk menyelesaikan persoalan program kuadratik digunakan metode complementary pivot. Untuk menguji keabsahan perhitungan ini, contoh persoalan program kuadratik pada bab III dihitung ulang dengan menggunakan program, kemudian hasil perhitungan program dibandingkan dengan hasil perhitungan dari contoh soal.

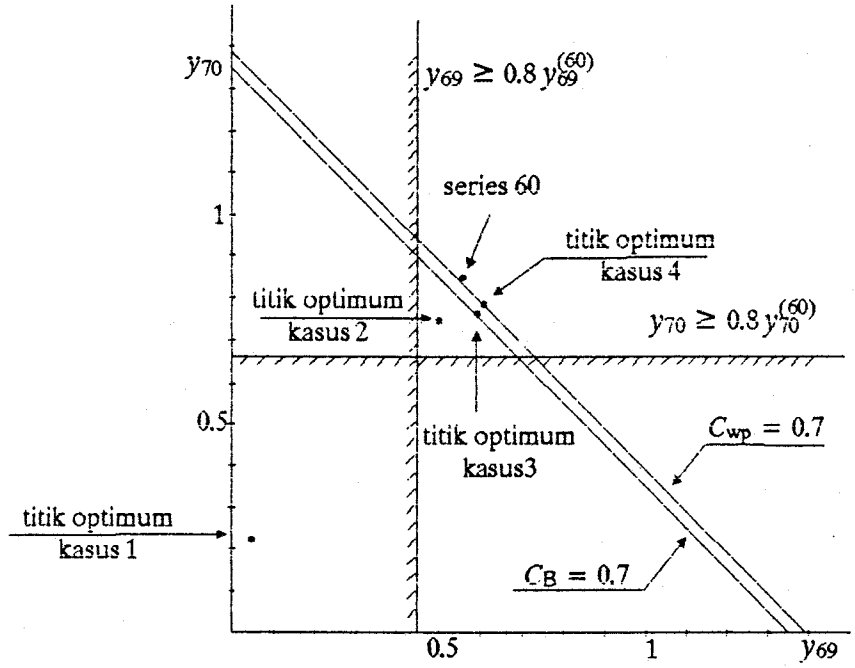
Setelah data-data dari contoh soal dimasukkan ke dalam program dan kemudian prosedur complementary pivot digunakan untuk menyelesaikannya, didapatkan hasil perhitungan sebagai berikut:

$$\dot{x}_1 = \frac{3}{2} \quad \text{dan} \quad \dot{x}_2 = \frac{1}{2}$$

sama seperti hasil perhitungan secara manual di dalam contoh soal.

☐ Perumusan program kuadratik

Pengujian yang terakhir adalah pengujian terhadap perumusan persoalan optimisasi ke dalam program kuadratik. Jika pada 2 pengujian sebelumnya telah memenuhi syarat, maka pengujian untuk tahap ini dapat dilakukan dengan memeriksa apakah hasil output program memenuhi semua



Gambar 5.1

persyaratan data constraints yang digunakan.

Gambar (5.1) adalah persoalan optimisasi untuk kapal series 60 block 60 untuk kecepatan F_n = 0.289 dan perbandingan sarat-lebar T/L = 0.053. Untuk memudahkan pengujian, semua offsets lambung dibuat tetap kecuali untuk offset ke 69 dan offset ke 70. Fungsi obyektif untuk persoalan tersebut adalah :

$$f(y) = 0.1586 + \begin{bmatrix} 0.02135 & -0.06149 \end{bmatrix} \begin{bmatrix} y_{69} \\ y_{70} \end{bmatrix}$$

$$+ \begin{bmatrix} y_{69} & y_{70} \end{bmatrix} \begin{bmatrix} 0.15418 & -0.08073 \\ -0.08073 & 0.15418 \end{bmatrix} \begin{bmatrix} y_{69} \\ y_{70} \end{bmatrix}$$

Data-data constraint yang digunakan adalah :

Constrain 1 : $y_{69} \geq 0.8\, y_{69}^{(60)}$, atau $y_{69} \geq 0.450$

Constraint 2 : $y_{70} \geq 0.8\, y_{70}^{(60)}$, atau $y_{70} \geq 0.673$

Constraint 3 : $C_{wp} = 0.7$, atau

$$0.1\, y_{69} + 0.1\, y_{70} = 0.139$$

Constraint 4 : CB = 0.59 , atau

$$0.0125\, y_{69} + 0.0125\, y_{70} = 0.0169$$

Dengan mengkombinasikan beberapa constraints di atas, akan diamati apakah hasil penyelesaian memenuhi constraints yang ditentukan. Hasil perhitungan untuk beberapa persoalan dengan kombinasi constraint yang berlainan dapat dilihat pada tabel (5.2).

Tabel 5.2

Kasus	constraints	y_{69}	y_{70}	C_w
1	-	0.048	0.225	0.1522
2	1, 2	0.501	0.746	0.1876
3	1, 2, 3	0.607	0.783	0.1981
4	1, 2, 4	0.589	0.765	0.1951
5	1, 2, 3, 4	tidak ada	tidak ada	-

Kasus 1 adalah persoalan optimisasi tanpa constraint, sedangkan pada kasus 2-5 digunakan kombinasi constraint yang berlainan. Dapat dilihat pada gambar (5.1), bahwa penyelesaian untuk kasus 1-4 berada pada daerah penyelesaian atau memenuhi persyaratan constraints yang ditentukan. Sedangkan untuk kasus 5 tidak didapatkan penyelesaian karena constraint 3 sejajar dengan constraint 4. Dari tabel (5.2) didapatkan bahwa semakin

banyak constraint yang digunakan maka nilai fungsi obyektif cenderung semakin besar atau penurunan tahanan gelombang semakin kecil. Juga dapat diamati bahwa, sesuai dengan logika, dengan menggunakan data constraint yang mendekati data series 60 maka harga offsets dan harga fungsi obyektif optimum mendekati data kapal series 60 block 60 yaitu :

$y_{69} = 0.553$

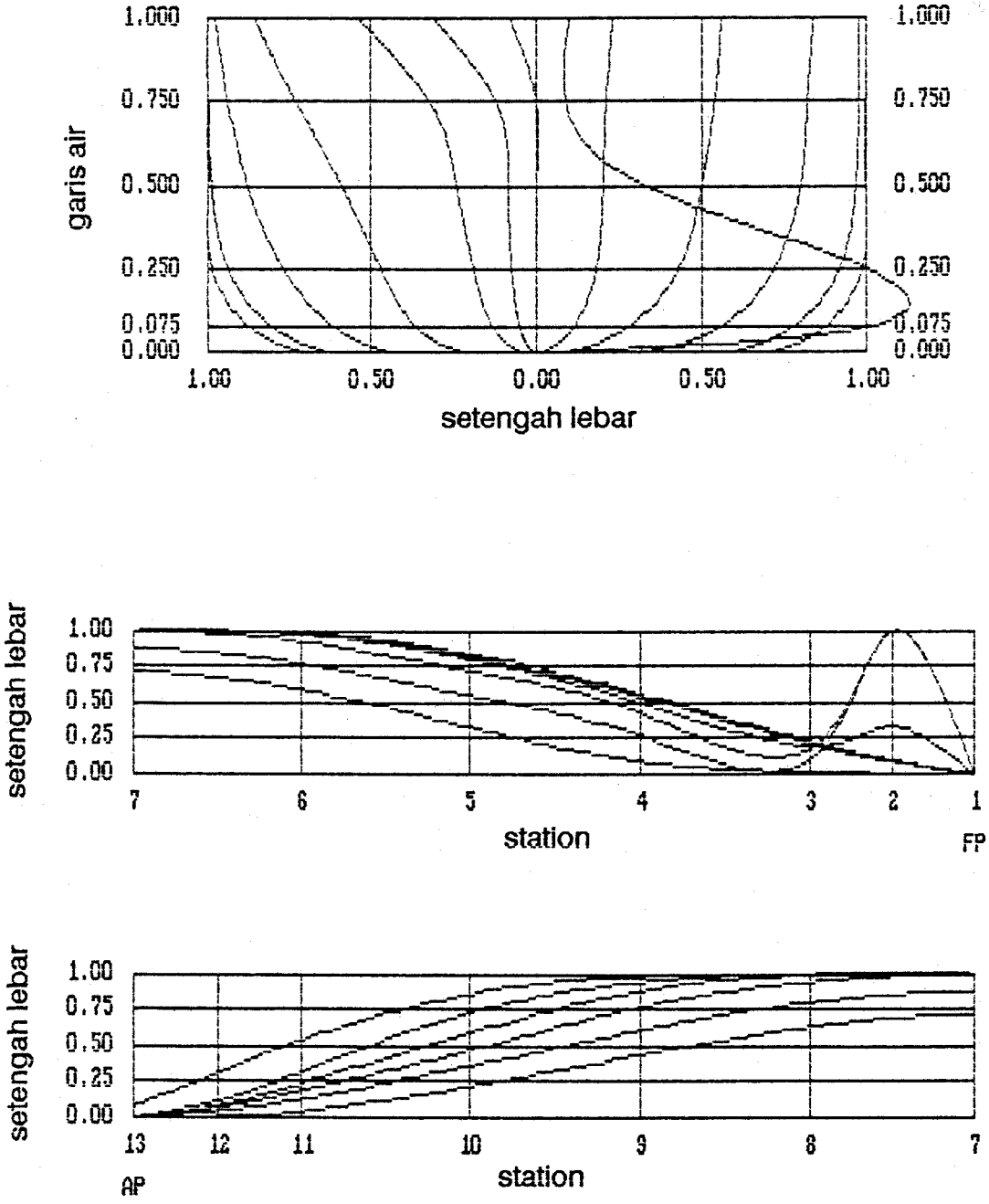
$y_{70} = 0.841$

$C_W = 0.203$

V.2. Analisa bentuk lambung optimum

Pada bagian ini hasil perhitungan program akan digunakan untuk melakukan analisa terhadap bentuk lambung kapal yang optimum untuk tahanan gelombang minimum. Perhitungan dilakukan terhadap kapal series 60 block 60 dengan data ukuran utama $T/L = 0.053$ dan $B/L = 0.1$. Sebagian besar perhitungan dilakukan untuk harga kecepatan $F_n = 0.289$, yaitu kecepatan standar kapal series 60. Pada kecepatan tersebut harga koefisien tahanan gelombang $C_W{}^{(60)} = 0.203$. Dengan mengubah kombinasi constraint yang digunakan akan diamati bagaimana bentuk lambung optimum serta besar penurunan tahanan gelombang yang dihasilkan.

Pada bagian pertama dilakukan perhitungan untuk kasus yang sangat sederhana. Semua offsets diambil sama seperti series 60 block 60, kecuali untuk station 2. Tangent line (WL 1) dan DWL dibuat tetap. Harga offsets tidak boleh kurang dari offsets semula dan tidak boleh melebihi lebar kapal, yaitu $y_{ij}{}^{(60)} \leq y_{ij} \leq 1$. Offsets lambung optimum yang didapat untuk station 2 menghasilkan bentuk semacam bulbous bow yang sangat lebar



Gambar 5.2

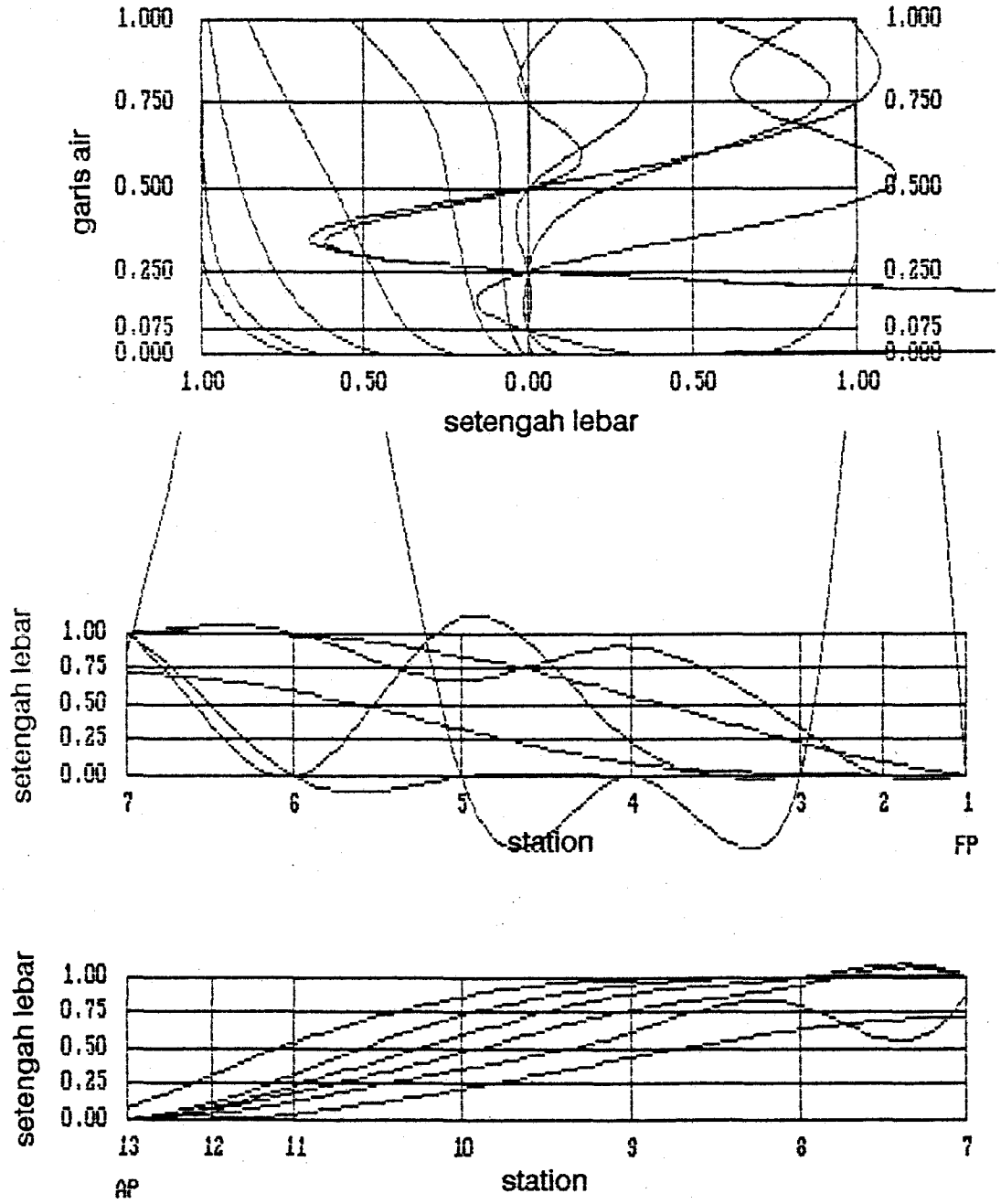
Rencana garis kapal optimum

seperti sayap (gambar 5.2). Hanya dengan mengubah satu station, koefisien tahanan gelombang yang dihasilkan C_W = 0.135, atau mengalami penurunan 35 % dari tahanan gelombang semula. Dari kasus pertama diperoleh gambaran tentang arti penting dari perancangan bentuk lambung di dalam menurunkan tahanan gelombang.

Selanjutnya gambar (5.3) - (5.5) akan dianalisa secara dekat. Pada ketiga kasus ini bagian belakang kapal adalah sama seperti series 60 block 60. Tangent line dan C_b dibuat tetap. Pada gambar (5.3) DWL tetap dan $y_{ij} \geq 0$, gambar (5.4) C_{WP} tetap dan $0 \leq y_{ij} \leq 1$, sedangkan gambar (5.5) C_{WP} tetap dan $0.7\, y_{ij}^{(60)} \leq y_{ij} \leq 1$. Koefisien tahanan gelombang untuk ketiga kasus tersebut berturut-turut adalah 0.074, 0.089, dan 0.092.

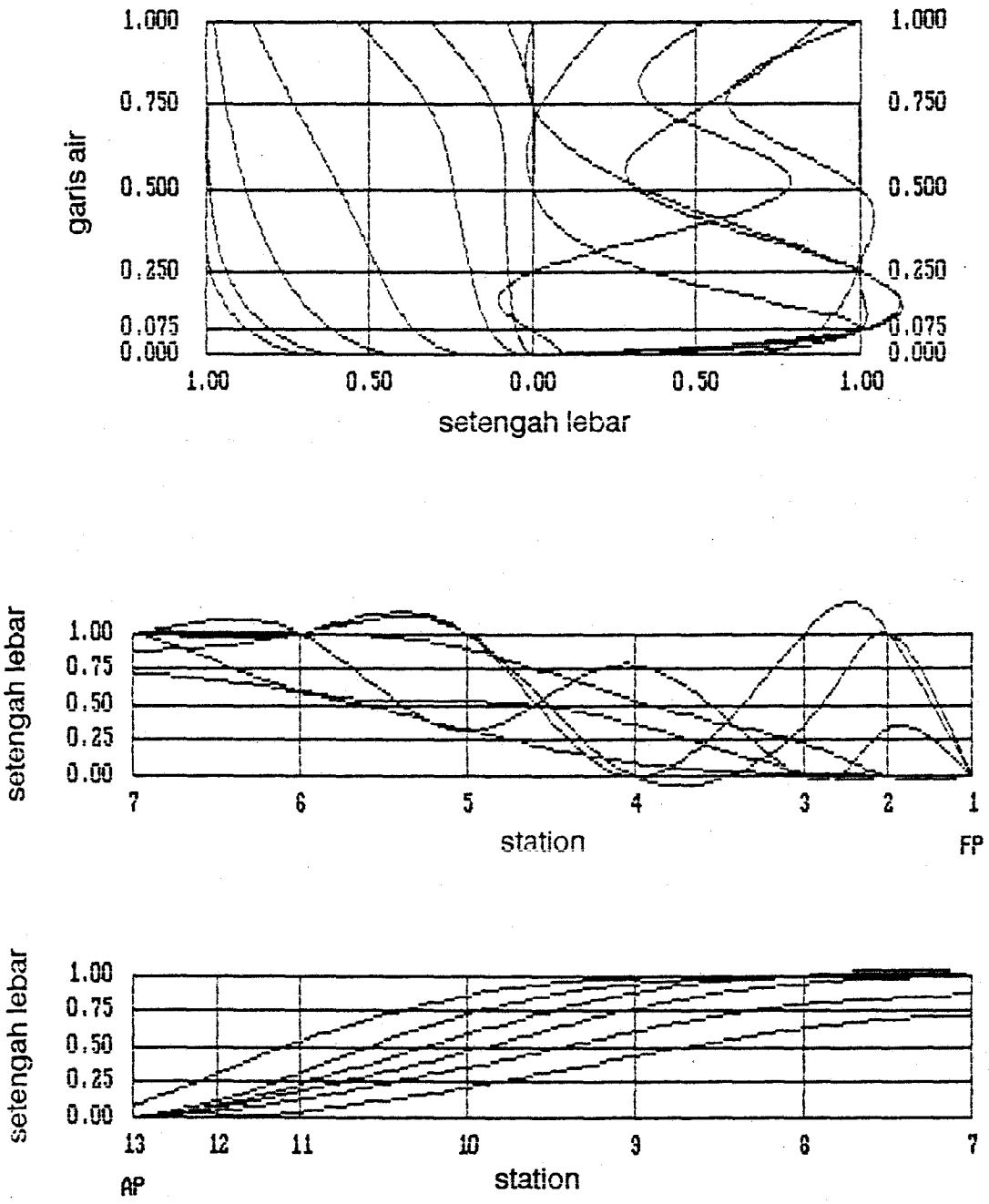
Perlu dicatat bahwa untuk menghasilkan suatu penyelesaian bentuk kapal beberapa constraint harus dipilih secara cermat. Jika tidak, penyelesaian yang dihasilkan mungkin tidak praktis atau bahkan tidak masuk akal. Bisa juga terjadi persoalan tanpa penyelesaian jika constraint yang dikenakan pada persoalan saling tidak bersesuaian.

Pada gambar (5.3) terdapat beberapa offsets yang bernilai negatif. Bentuk semacam ini sudah barang tentu tidak akan pernah ada. Walaupun persoalan program kuadratik standar mengharuskan nilai offset adalah lebih besar atau sama dengan nol, tetapi pengaruh interpolasi antar offsets memungkinkan offsets bernilai negatip. Juga terlihat beberapa offsets yang bernilai terlalu besar melebihi lebar kapal, sehingga tampak tidak praktis. Untuk itu, tampaknya perlu dilakukan penentuan batas atas dan batas bawah dari offsets secara cermat. Walaupun semua bentuk diatas adalah tidak praktis sebagaimana layaknya kapal, tetapi dari sini bisa diperoleh gambaran



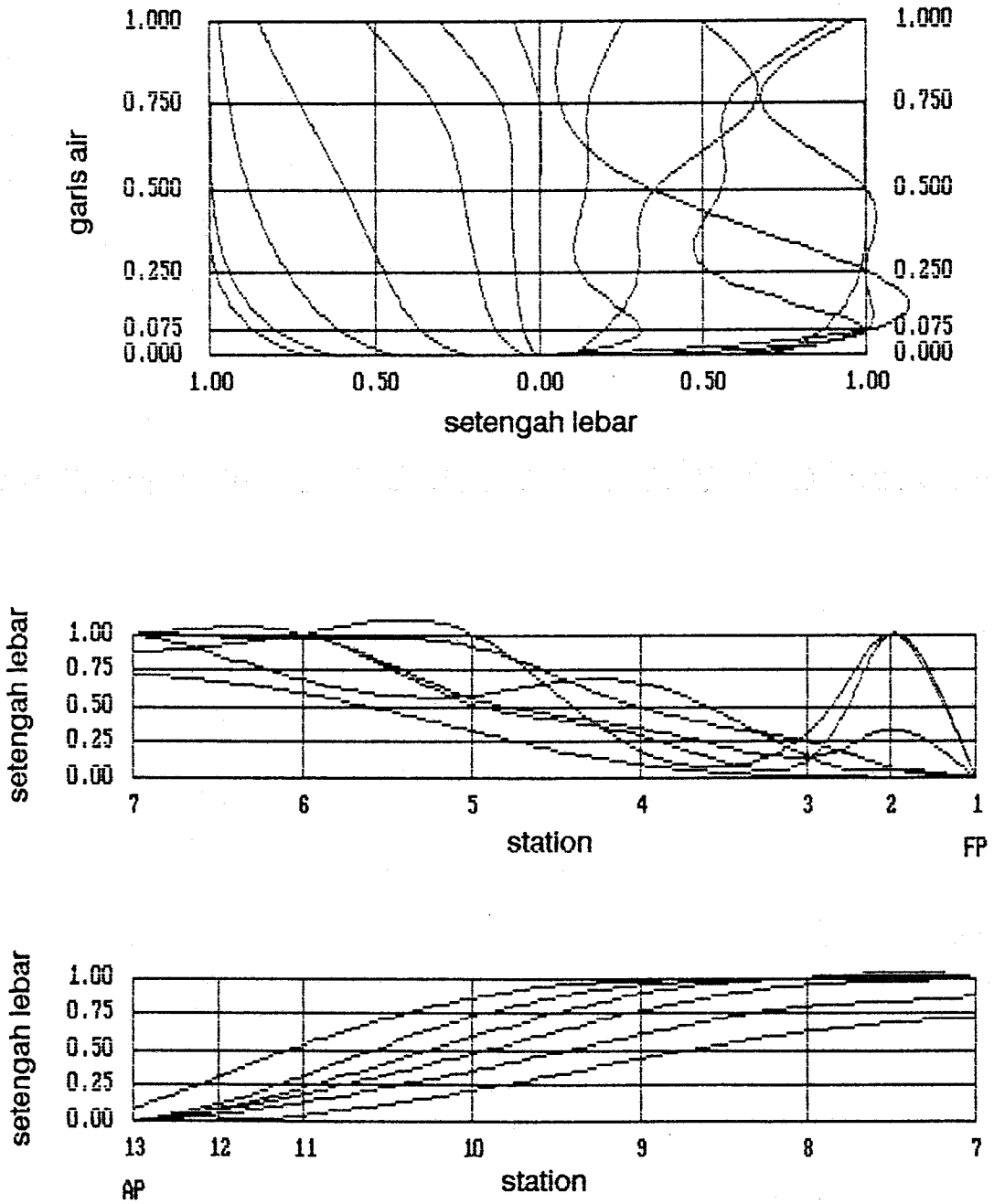
Gambar 5.3

Rencana garis kapal optimum



Gambar 5.4

Rencana garis kapal optimum



Gambar 5.5

Rencana garis kapal optimum

penting dari bentuk optimal kapal, khususnya bulbous bow dan lambung bergelombang.

Selanjutnya akan dicoba bagaimana kalau offsets hanya diijinkan untuk range yang sempit, barangkali akan diperoleh bentuk yang lebih baik. Pada gambar (5.6) - (5.9) semua bagian belakang dibuat sama seperti series 60 block 60, C_{wp} dan tangent line tetap, sedangkan constraint lainnya adalah

untuk gambar (5.6) $0.8\, y_{ij}^{(60)} \le y_{ij} \le y_{ij}^{(60)}$, $C_b \le 0.591$,

untuk gambar (5.7) $y_{ij}^{(60)} \le y_{ij} \le 1.2\, y_{ij}^{(60)}$, $C_b \ge 0.591$,

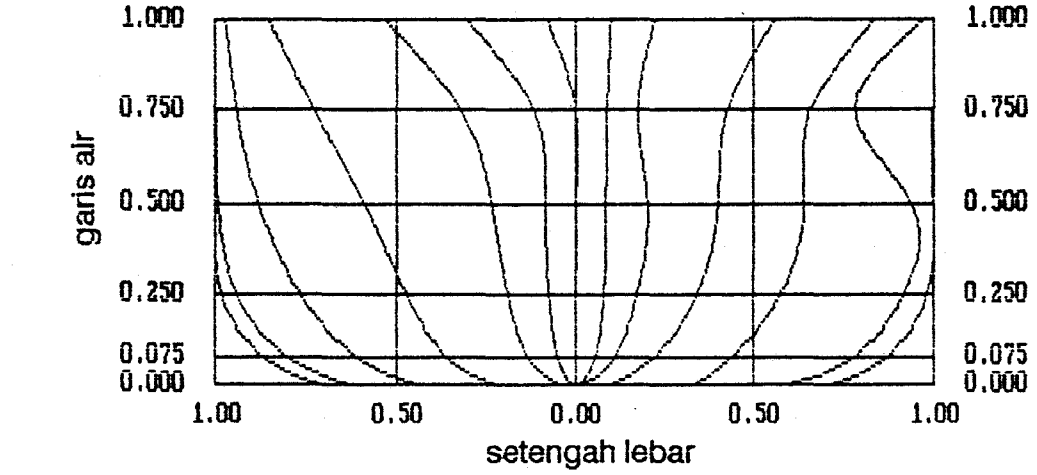
$y_{ij} \le 1$ untuk station 5 dan 6

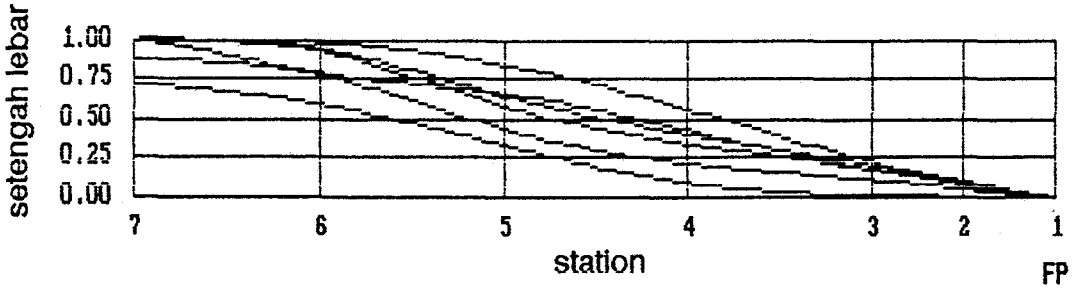
untuk gambar (5.8) $0.8\, y_{ij}^{(60)} \le y_{ij} \le 1.2\, y_{ij}^{(60)}$, $C_b = 0.591$,

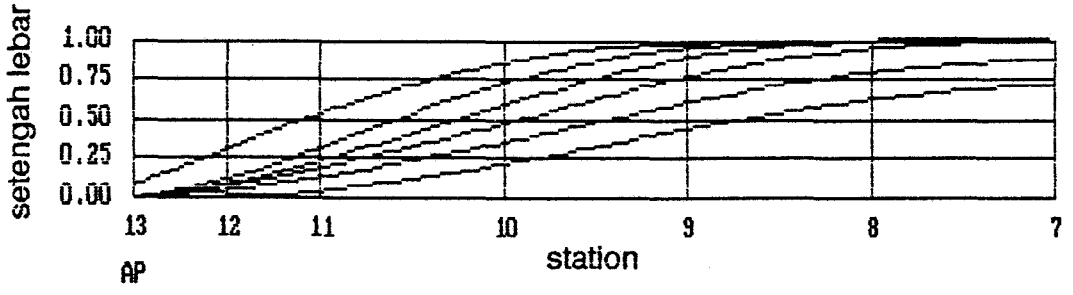
untuk gambar (5.9) $0.9\, y_{ij}^{(60)} \le y_{ij} \le 1.1\, y_{ij}^{(60)}$, $C_b = 0.591$

Semua bentuk lambung pada gambar (5.6) - (5.9) adalah masuk akal dan lebih baik dari hasil perhitungan sebelumnya. Masing-masing mempunyai harga koefisien tahanan gelombang 0.116, 0.189, 0.147, dan 0.164. Karena range yang sempit pada daerah haluan, maka bentuk bulb tidak mungkin terjadi. Tetapi terlihat bahwa bulb ditekan menuju ke belakang membentuk bermacam-macam lambung bergelombang.

Selanjutnya dilakukan analisa tentang pengaruh penambahan displasemen terhadap bentuk optimal. Gambar (5.10) - (5.12) adalah kasus untuk penambahan displasemen yang berbeda-beda. Semuanya memiliki bagian belakang yang sama seperti series 60 block 60, C_{wp} dan tangent line tetap. Offsets series 60 dipakai sebagai batas bawah, yaitu $y_{ij} \ge y_{ij}^{(60)}$, dan $y_{ij} \le 1.1$. Sudut section pada station 2, 3 dan 4 diambil lebih besar atau sama dengan 10 derajat untuk mengurangi kemungkinan slamming pada daerah haluan. Tambahan displasemen untuk gambar (5.10) - (5.12) masing-masing

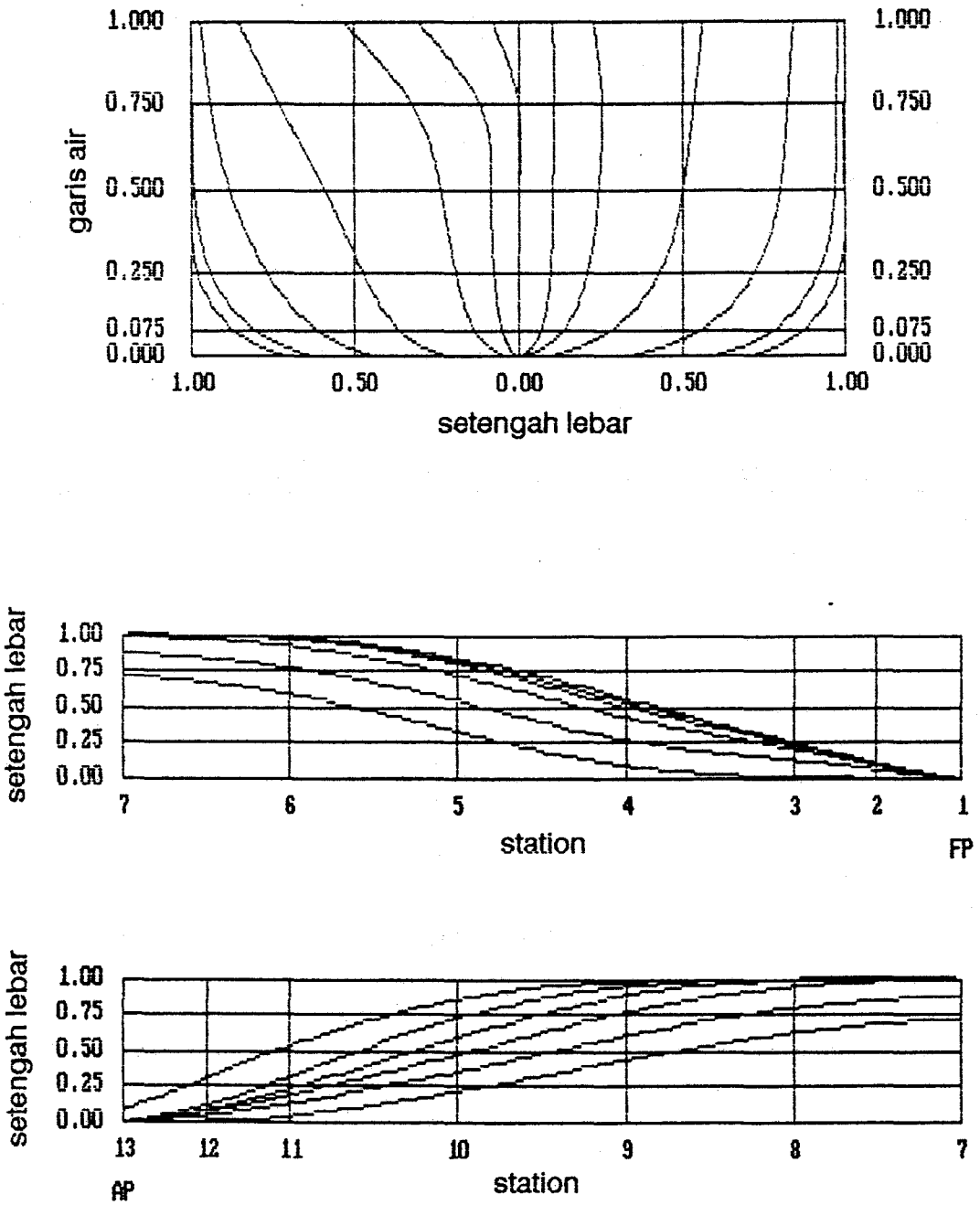






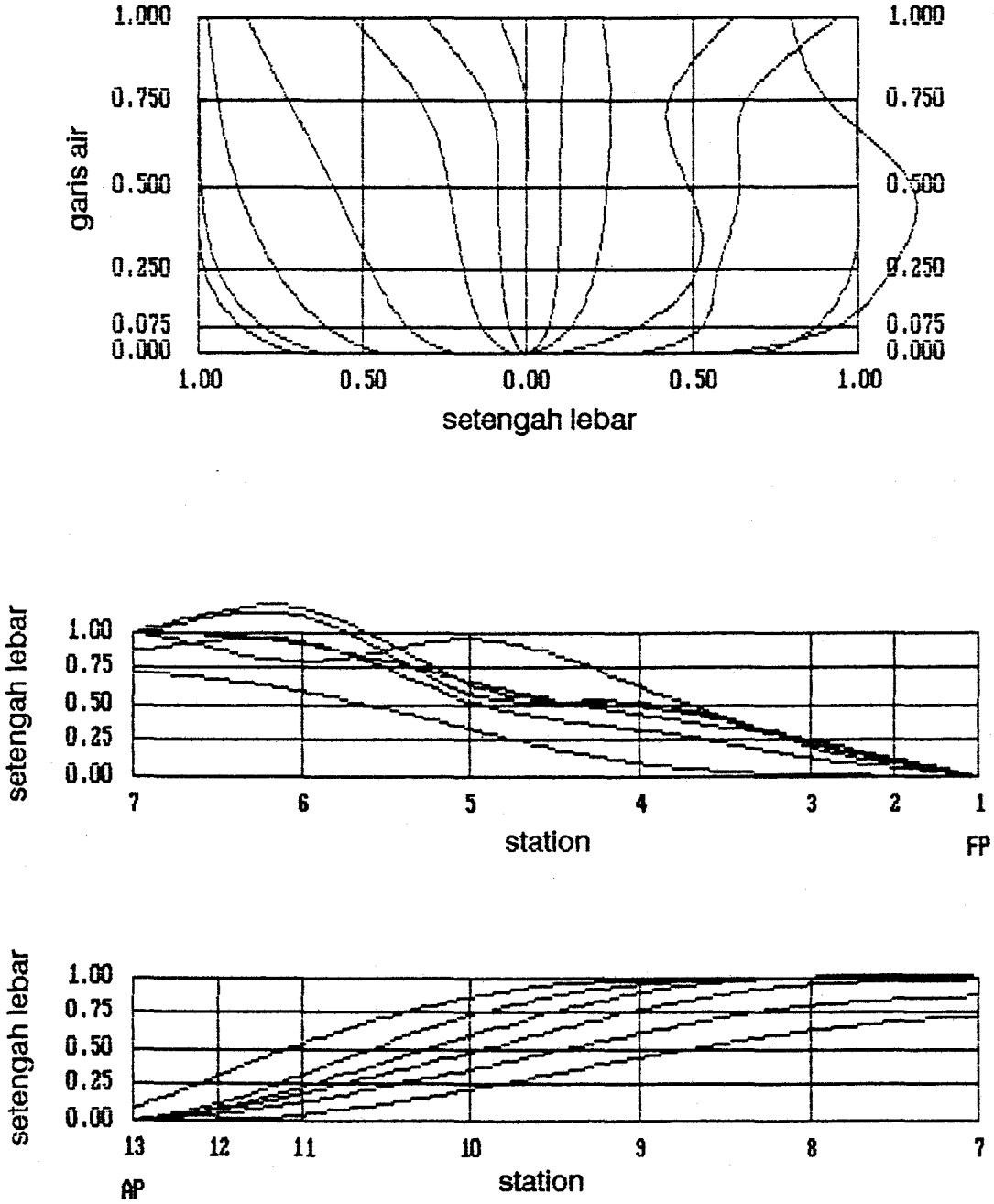
Gambar 5.6

Rencana garis kapal optimum



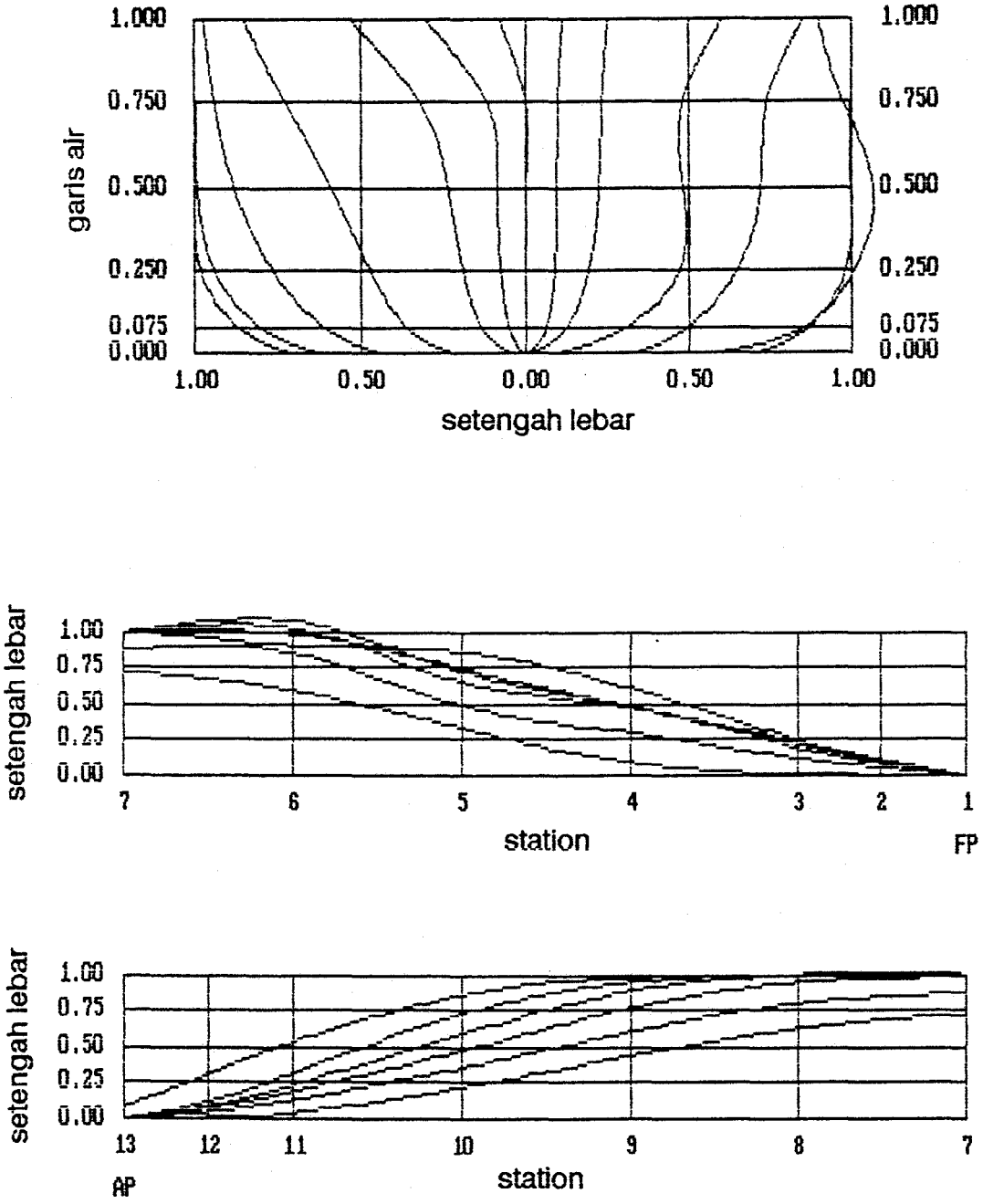
Gambar 5.7

Rencana garis kapal optimum



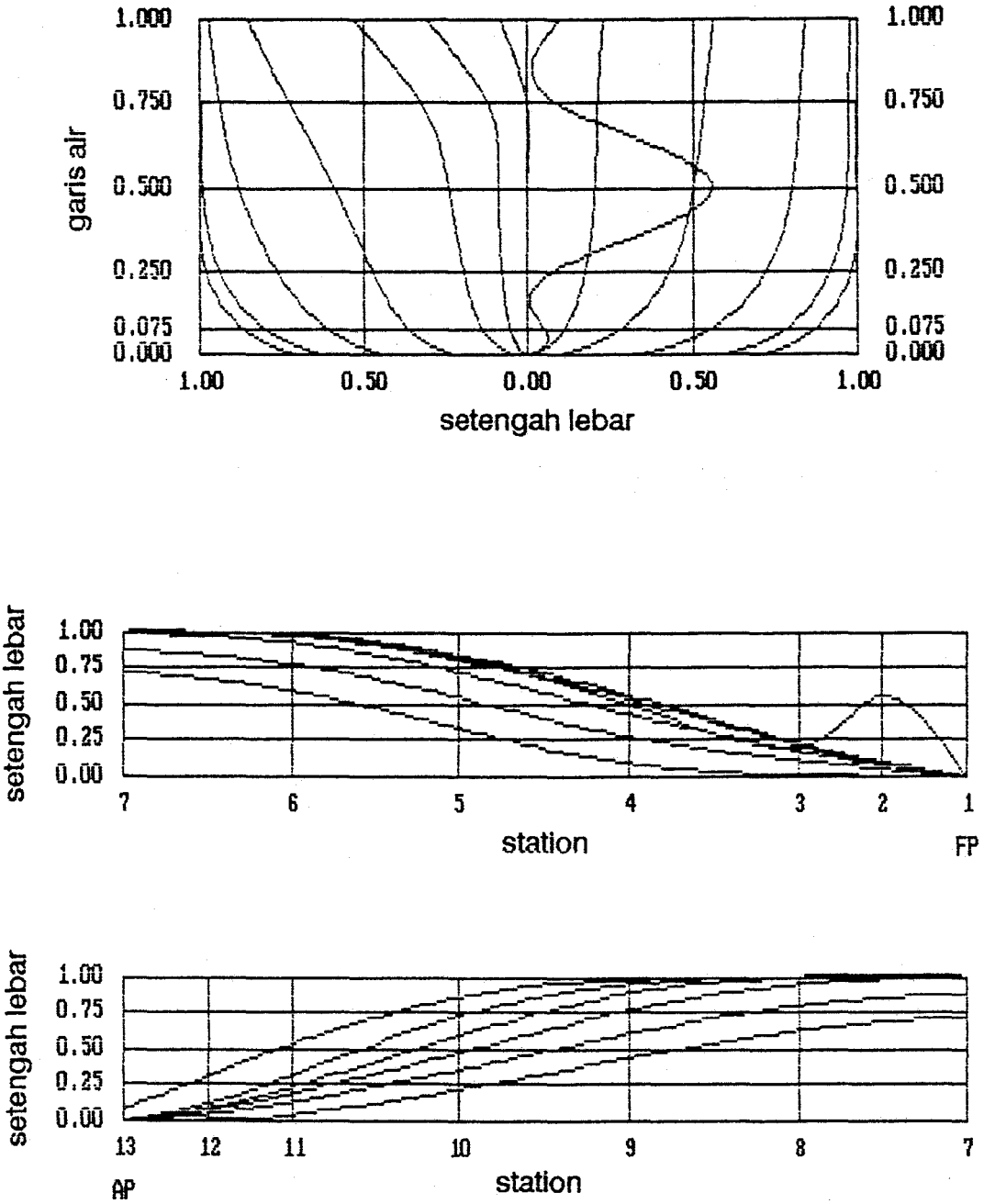
Gambar 5.8

Rencana garis kapal optimum



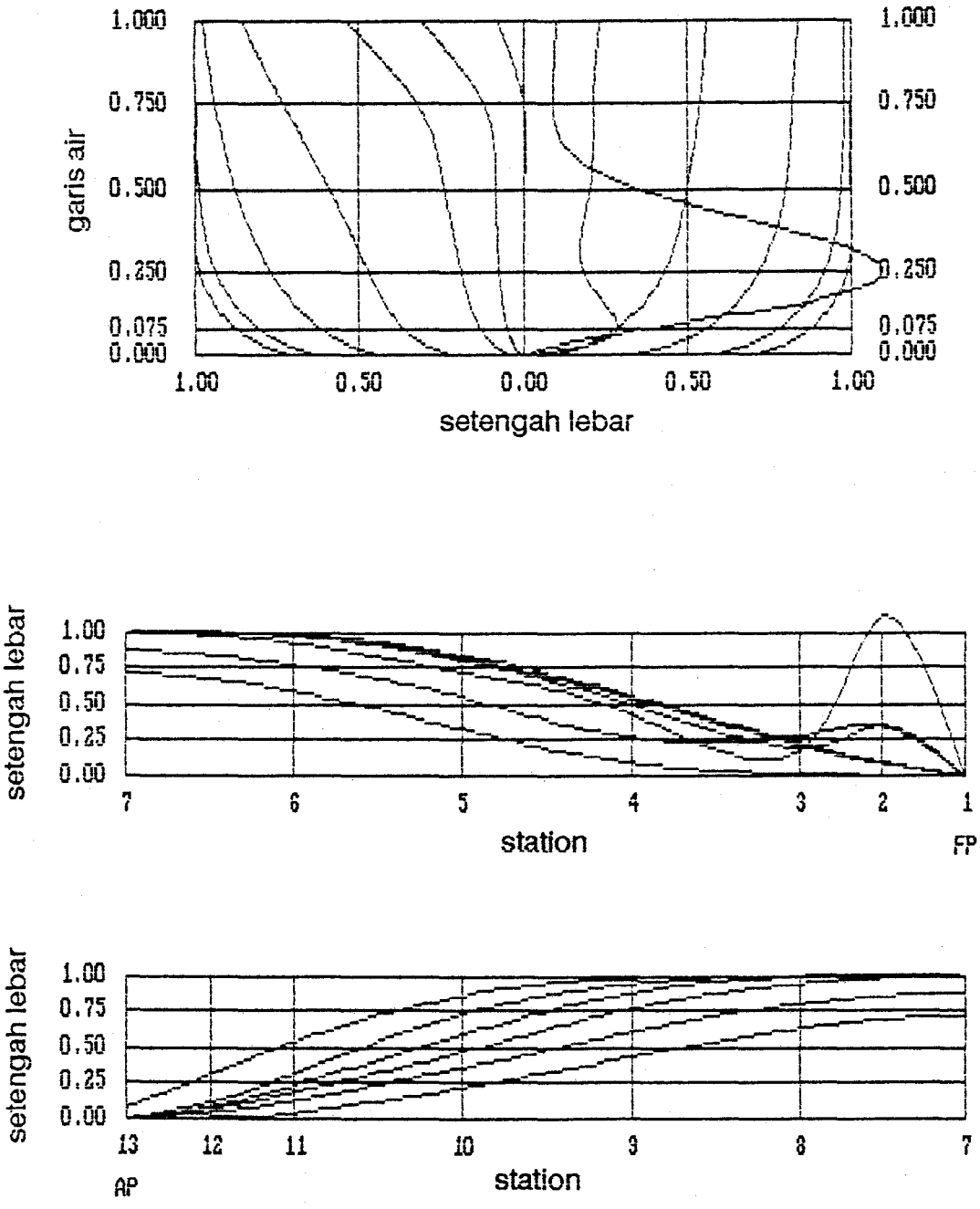
Gambar 5.9

Rencana garis kapal optimum



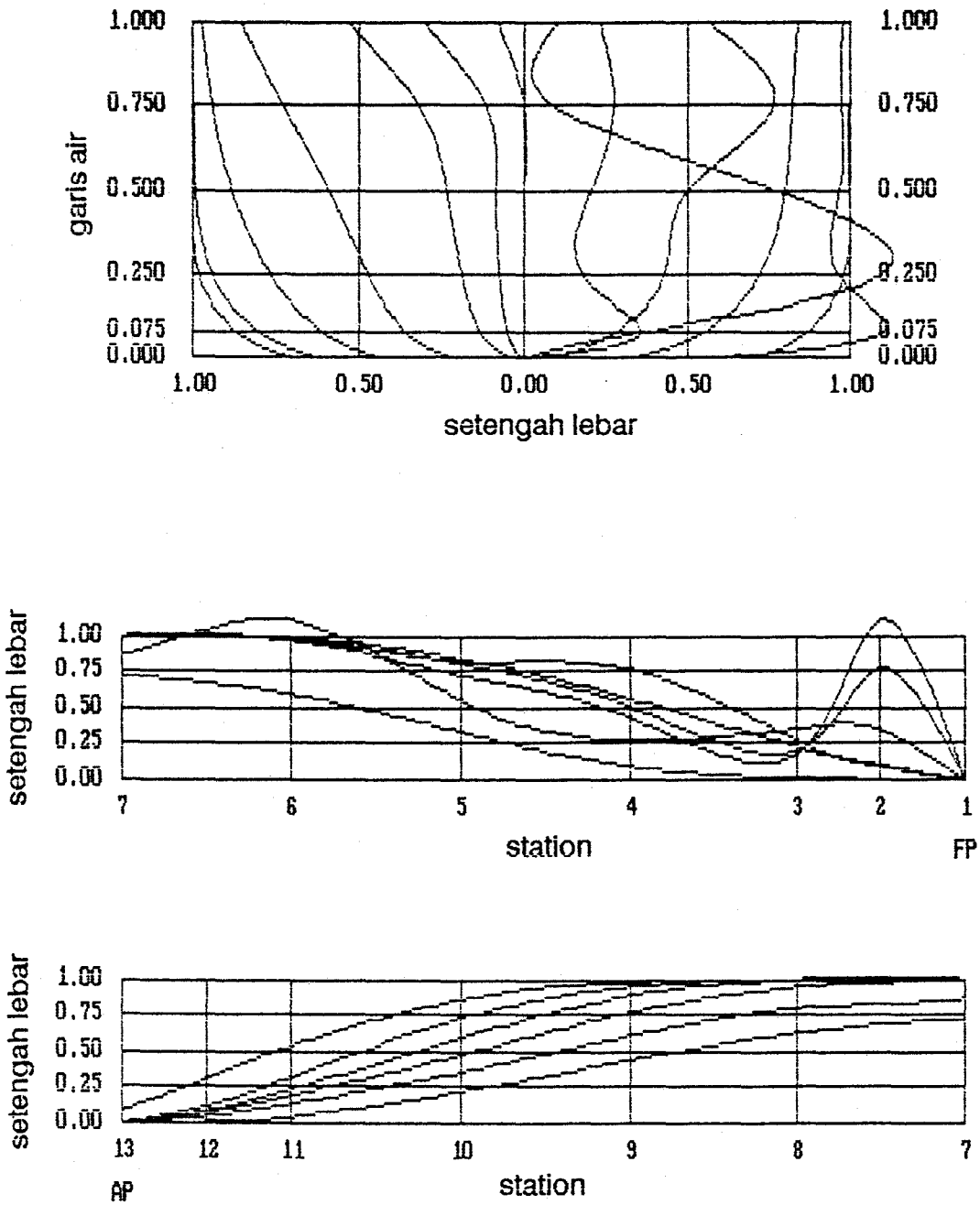
Gambar 5.10

Rencana garis kapal optimum



Gambar 5.11

Rencana garis kapal optimum



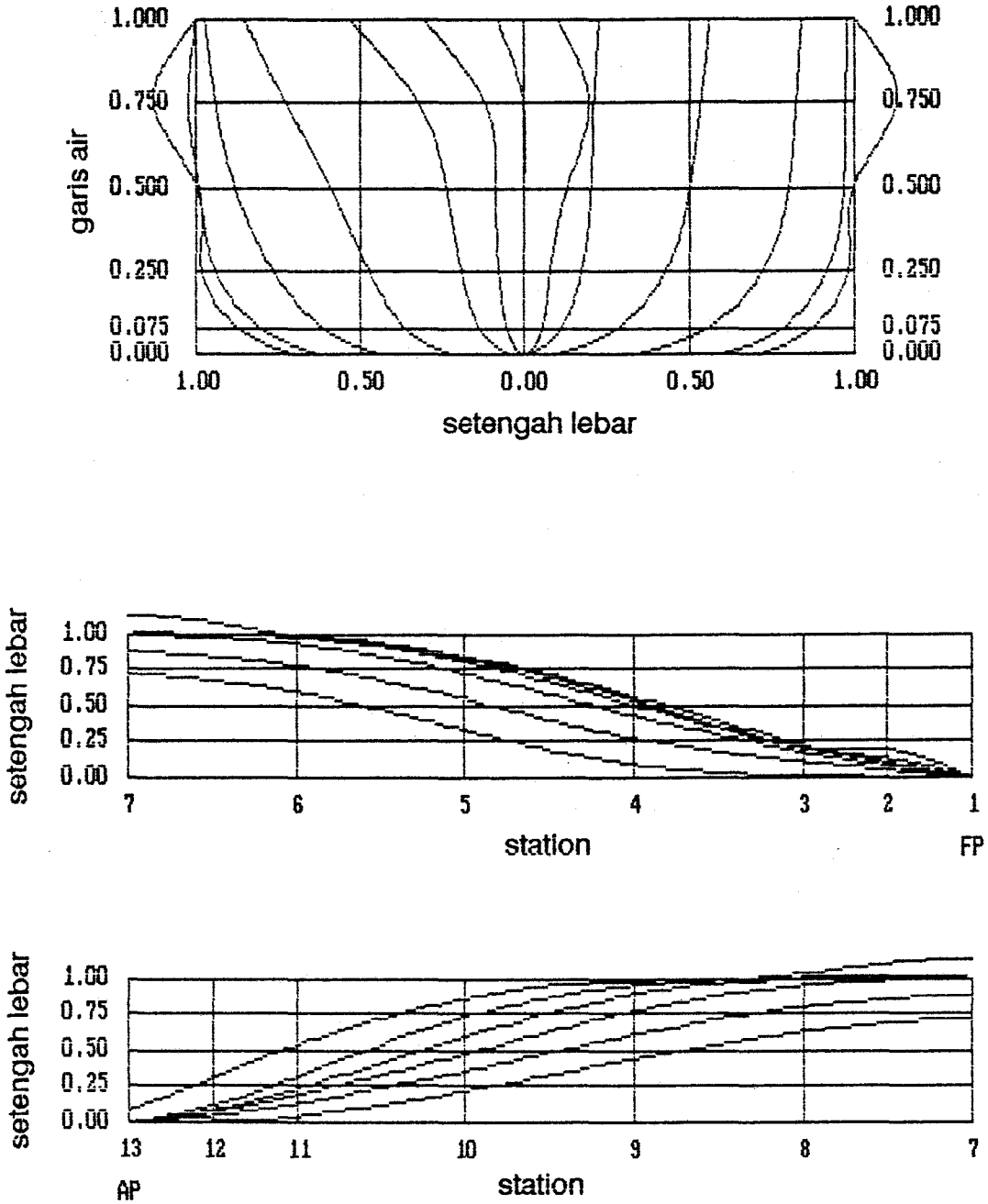
Gambar 5.12

Rencana garis kapal optimum

adalah 1%, 3%, dan 6%. Untuk penambahan displasemen yang kecil terlihat bahwa tambahan displasemen terdistribusi pada daerah depan membentuk semacam bulb. Sedangkan untuk penambahan yang besar distribusi volume lebih merata ke belakang membentuk lambung bergelombang. Harga koefisien tahanan gelombang yang dihasilkan untuk keempat kasus tersebut berturut-turut adalah 0.165, 0.138, dan 0.141.

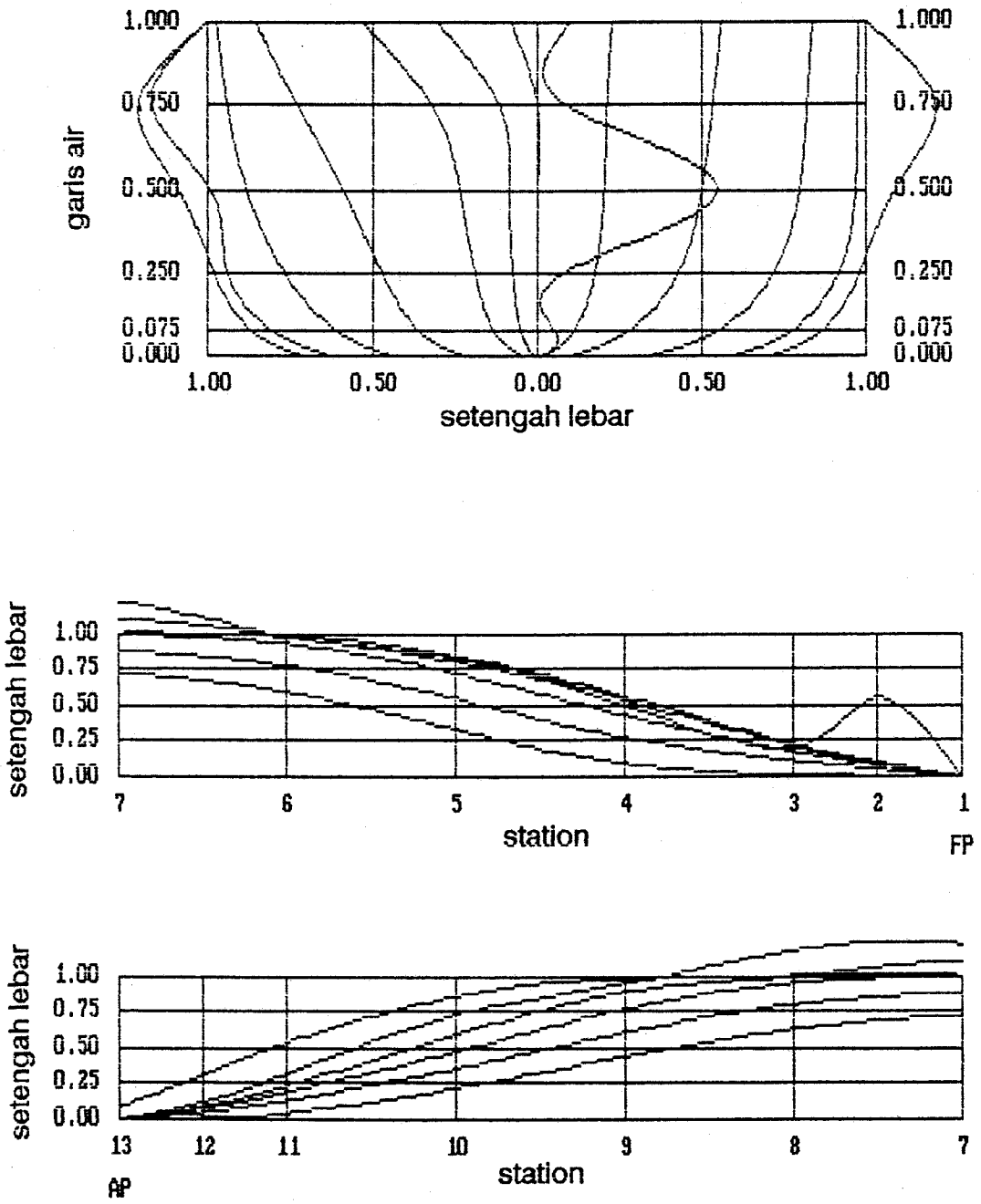
Serupa dengan kasus sebelumnya, pengaruh penambahan displasemen akan diamati untuk seluruh bagian lambung kapal kecuali untuk station 1,11,12, dan 13. Semua constraints sama seperti sebelumnya kecuali batas atas offsets dihilangkan. Hasil perhitungan menunjukkan bahwa tambahan displasemen terpusat pada daerah haluan dan midship (gambar 5.13-5.15). Nilai koefisien tahanan gelombang untuk bentuk diatas berturut-turut adalah 0.160, 0.108, dan 0.071. Semakin besar penambahan displasemen, bulb atau tonjolan lambung yang dihasilkan mempunyai offsets maksimum yang semakin rendah. Juga dapat diamati bahwa bulb yang dihasilkan didaerah midship cenderung lebih tinggi daripada bulb pada daerah haluan.

Pada kasus berikutnya akan dianalisa pengaruh perubahan kecepatan terhadap bentuk optimum kapal. Perhitungan dilakukan untuk harga bilangan Froude 0.20, 0.25, 0.30, dan 0.35. Constraints yang digunakan sama seperti kasus sebelumnya kecuali untuk penambahan displasemen, yaitu 2%. Nilai koefisien tahanan gelombang pada kecepatan tersebut berturut-turut adalah 0.016, 0.035, 0.193, dan 0.254. Sedangkan koefisien tahanan untuk bentuk optimum adalah 0.007, 0.026, 0.126, dan 0.207. Hasil perhitungan menunjukkan bahwa pada kecepatan rendah



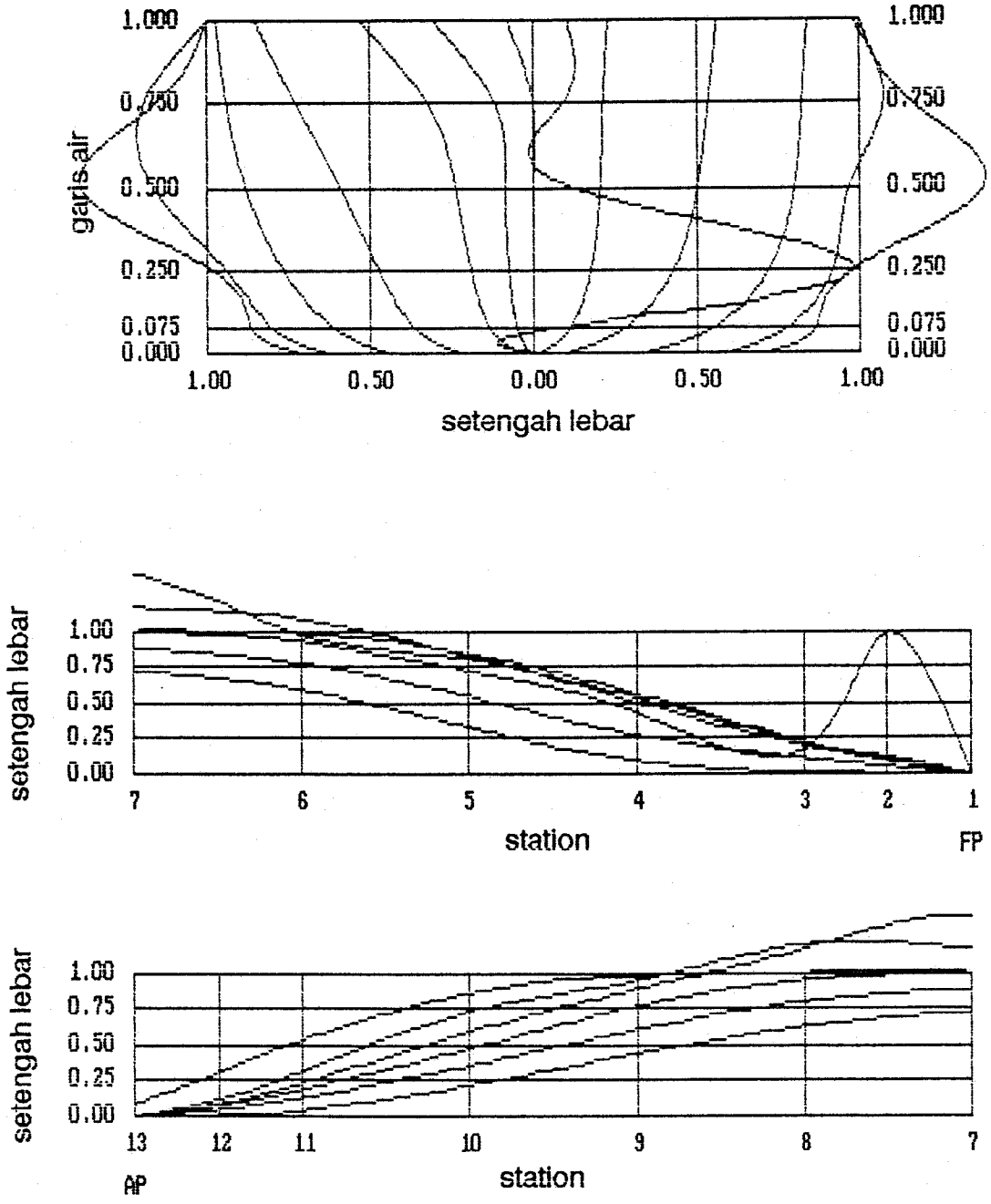
Gambar 5.13

Rencana garis kapal optimum



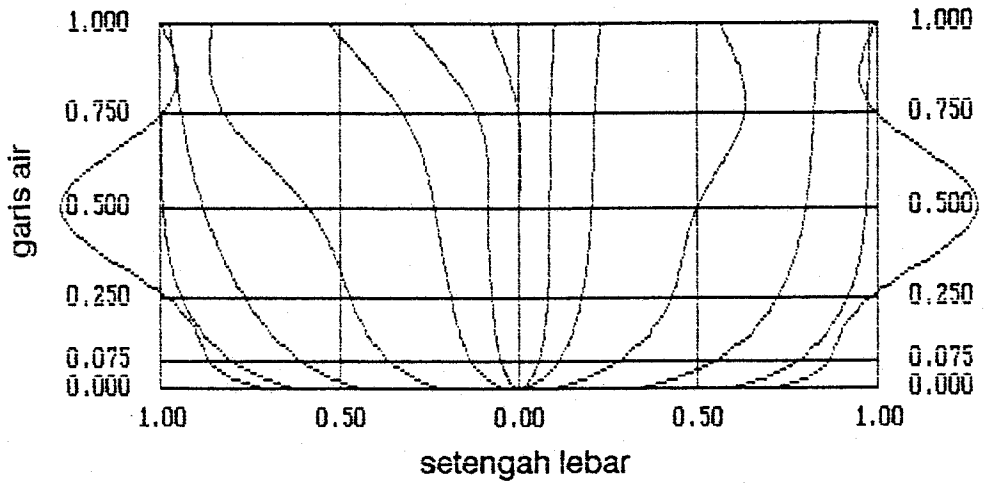
Gambar 5.14

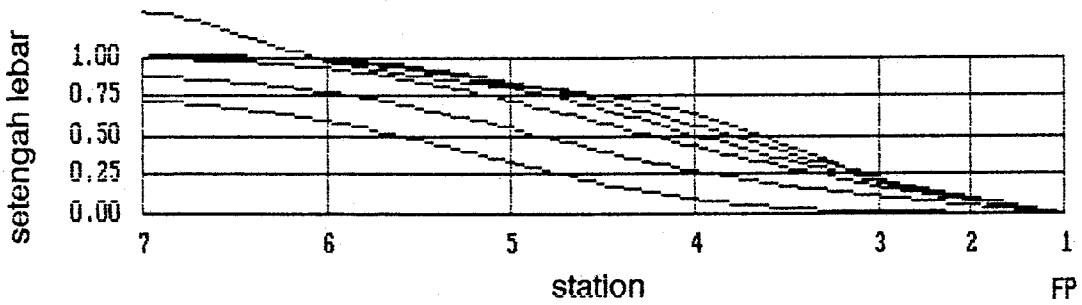
Rencana garis kapal optimum

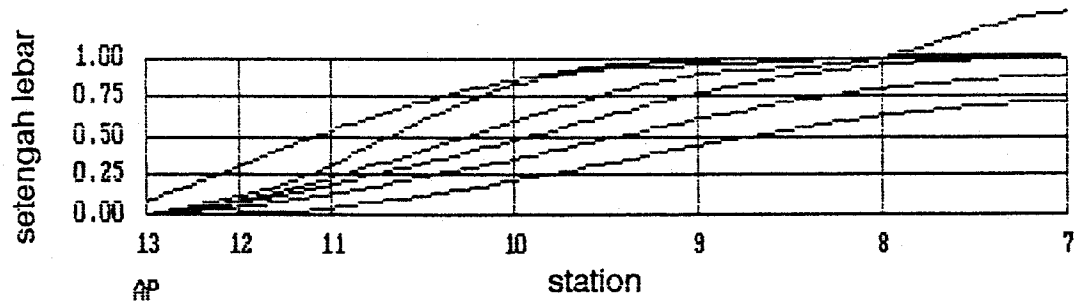


Gambar 5.15

Rencana garis kapal optimum

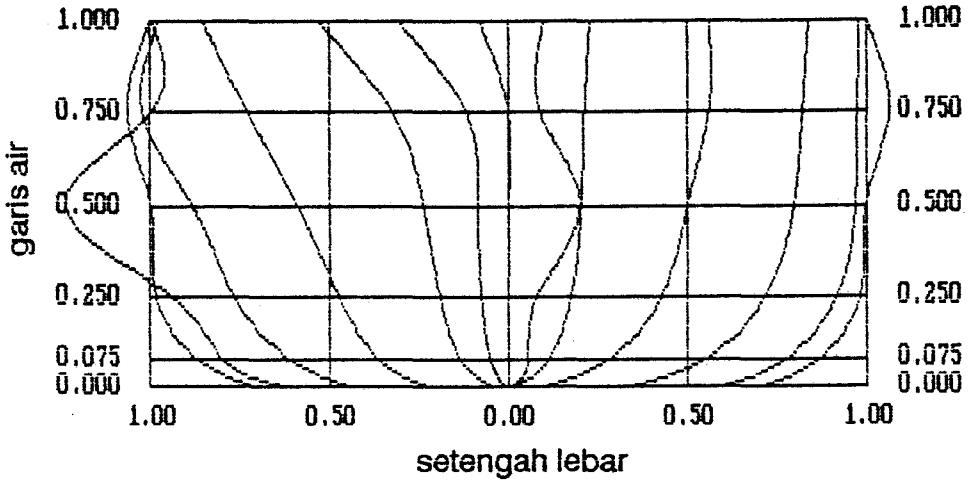


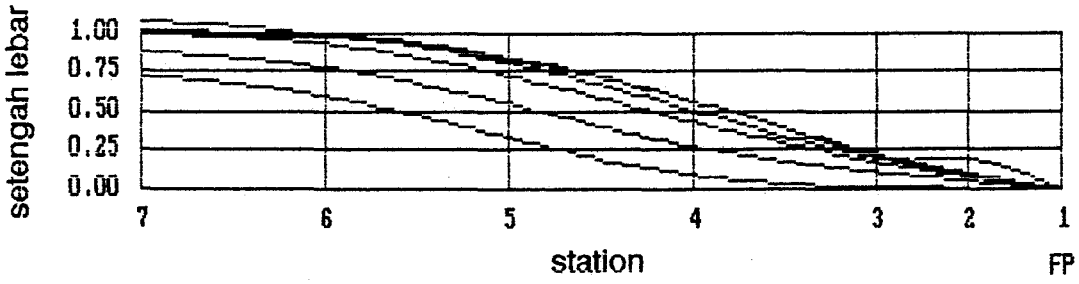


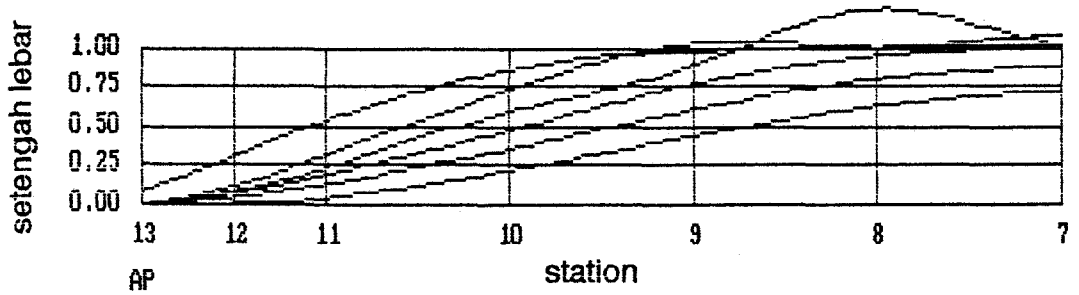


Gambar 5.16

Rencana garis kapal optimum

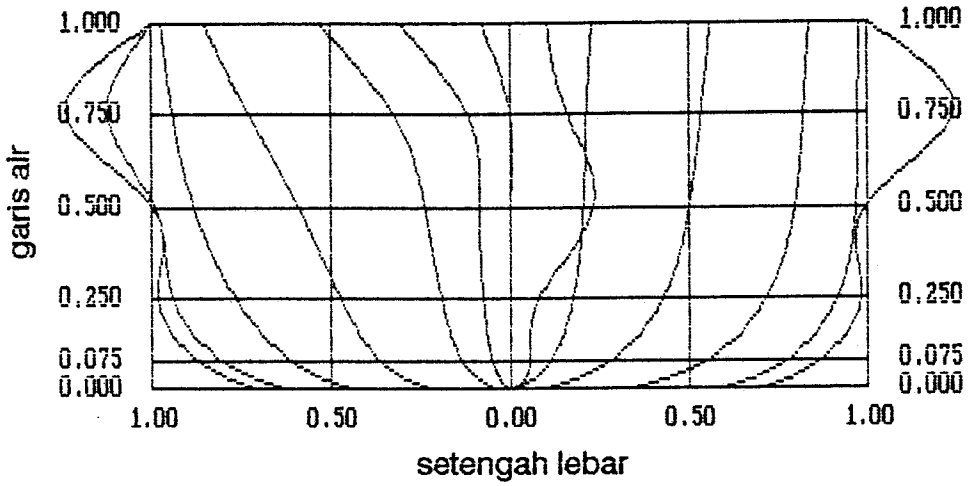


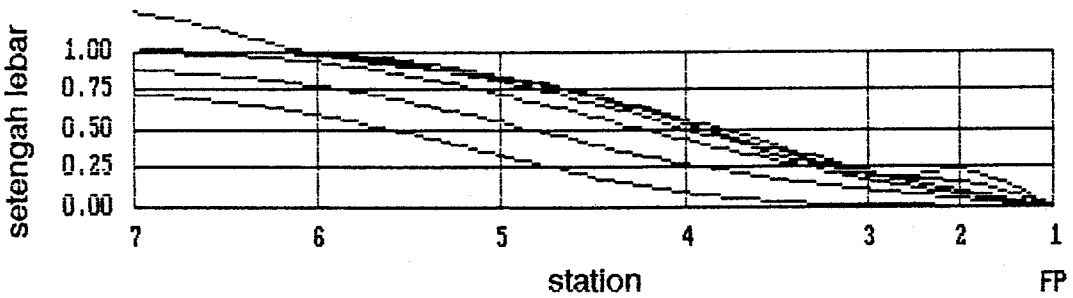


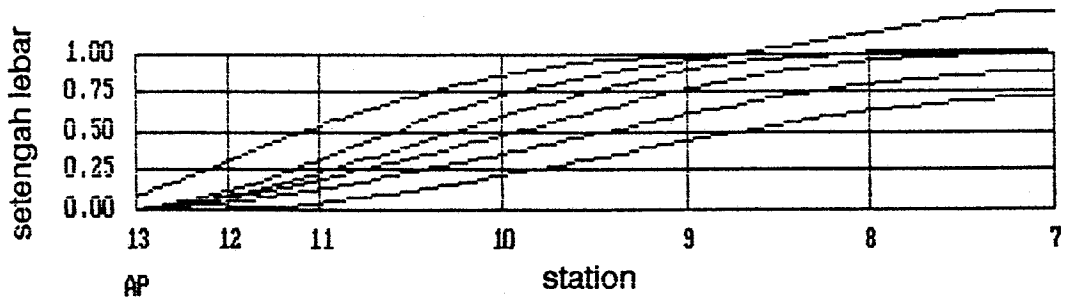


Gambar 5.17

Rencana garis kapal optimum

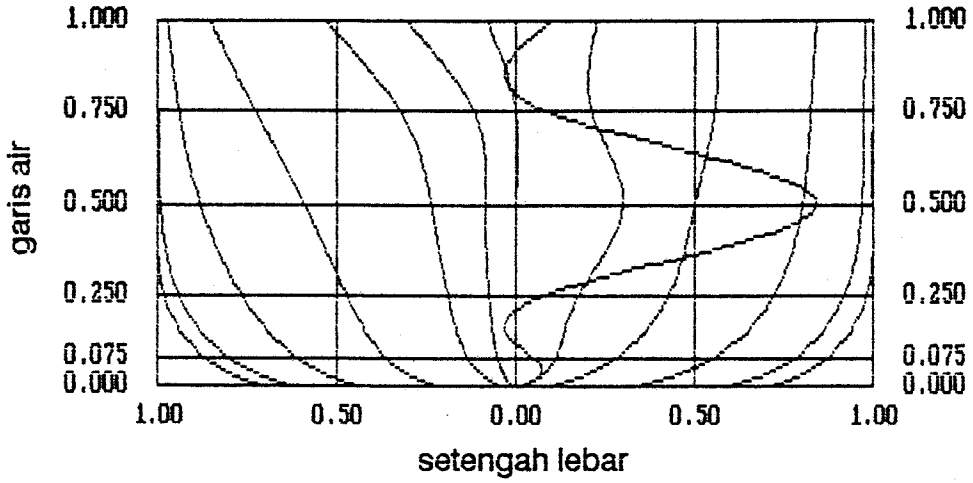


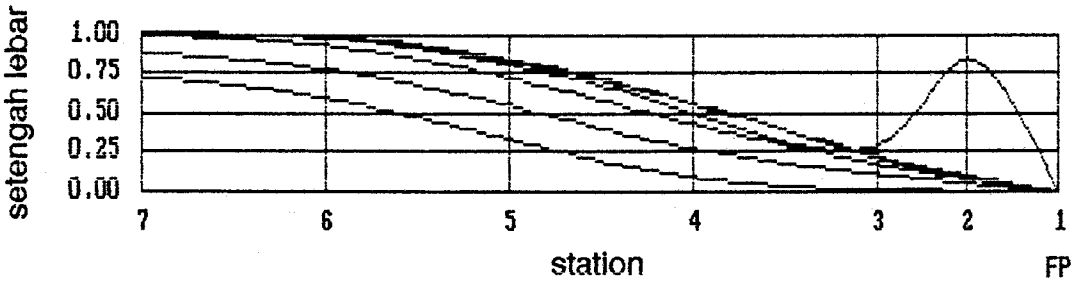


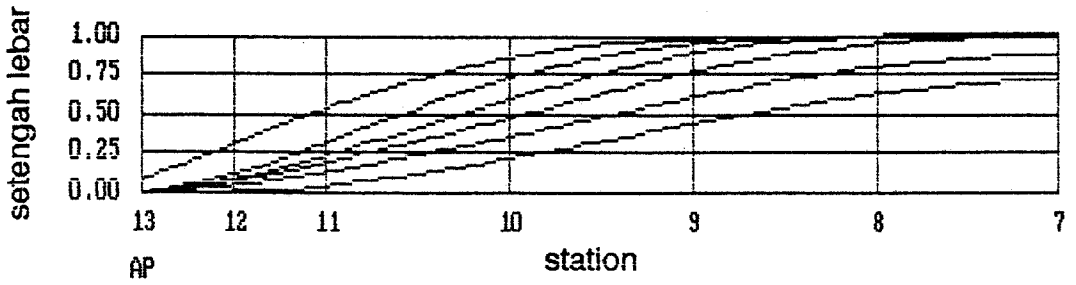


Gambar 5.18

Rencana garis kapal optimum







Gambar 5.19

Rencana garis kapal optimum

tambahan displasemen cenderung merata membentuk lambung bergelombang, selanjutnya jika kecepatan dinaikkan tambahan displasemen semakin terpusat pada daerah haluan dan daerah midship membentuk bulb, dan akhirnya pada kecepatan yang sangat tinggi tambahan displasenen terpusat di daerah haluan (gambar 5.16-5.19).

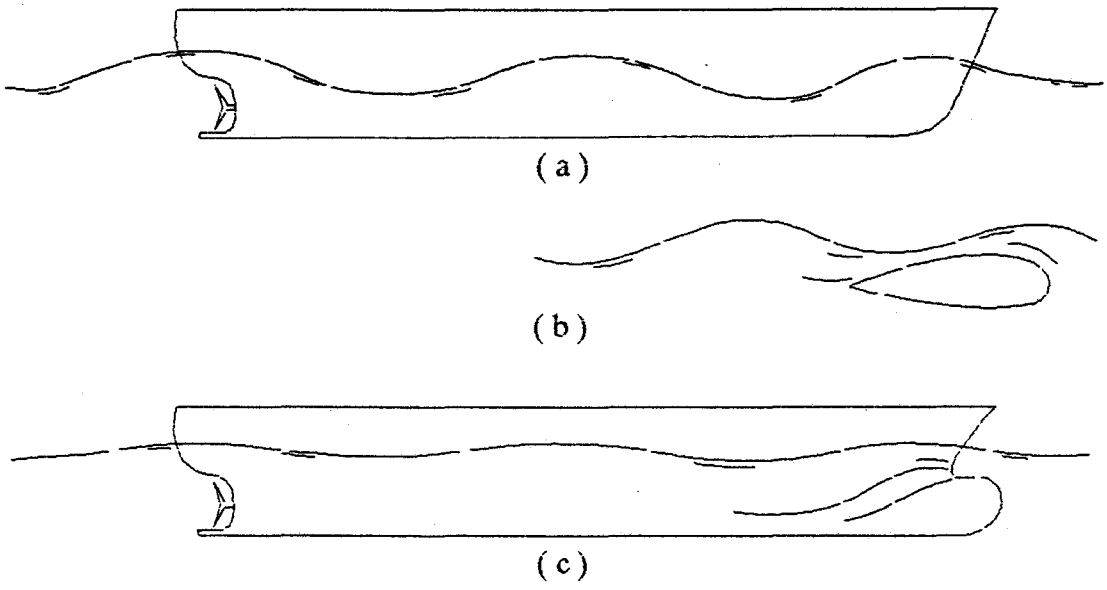
Khusus mengenai bulbous bow, hasil dari perhitungan (gambar 5.2, 5.10, 5.11, dan 5.19) menunjukkan adanya beberapa kesesuaian dengan apa yang dikemukakan Wigley tentang bulbous bow [7], antara lain :

- Dalam arah longitudinal, bulb harus sependek mungkin.
- Dalam arah lateral, bulb harus selebar mungkin.
- Puncak dari bulb tidak boleh terlalu dekat dengan permukaan air

Bagaimana bentuk-bentuk di atas menghasilkan tahanan gelombang minimum, dapat dijelaskan sebagai berikut. Menurut teori tahanan gelombang, tahanan gelombang kapal di hasilkan oleh interferensi dari beberapa sistim gelombang individual

$$R_W = V^6 \text{ (bentuk konstan } + \text{ 4 bentuk berosilasi)}$$

Dengan demikian, persoalan optimisasi bentuk lambung bisa diartikan sebagai persoalan mencari bentuk lambung kapal sedemikian sehingga hasil penjumlahan komponen tahanan gelombang pada persamaan di atas adalah yang paling minimum, yang berarti hasil interferensi beberapa gelombang inividual adalah yang paling minimum. Dengan menggunakan kaidah matematika yang ada dapat diketahui titik-titik optimum (atau yang terdekat dengan titik-titik optimum) dari persamaan tahanan gelombang. Jadi, bentuk-bentuk optimum di atas adalah bentuk lambung yang pada kondisi



Gambar 5.20 Pengaruh bulbousbow
terhadap sistim gelombang kapal

sarat, kecepatan, dan constraints tertentu menghasilkan interferensi gelombang yang paling minimum.

Sebagai gambaran bagaimana bentuk-bentuk tersebut menurunkan tahanan gelombang, diberikan ilustrasi untuk kasus yang sangat umum, yaitu bulbous bow. Gambar (5.20.a) menjelaskan sistim gelombang kapal tanpa bulb. Gambar (5.20.b) menjelaskan sistim gelombang dari bulb. Jika digabungkan, kedua sistim gelombang tersebut akan saling mengganggu satu sama lain dan menurunkan elevasi gelombang, sehingga tahanan gelombang menjadi lebih kecil (gambar 5.20.c). Serupa dengan kasus bulbousbow, penurunan tahanan gelombang pada lambung bergelombang dapat dijelaskan dengan cara yang sama. Bentuk lambung yang bergelombang mampu membangkitkan sistim gelombang baru yang akan

mereduksi sistim gelombang utama kapal.

Dari beberapa kasus di atas, terlihat bahwa pemilihan constraints menentukan hasil optimisasi. Tergantung dari constraint yang digunakan, semakin sedikit constraint yang digunakan akan dihasilkan penurunan tahanan gelombang yang lebih besar, tetapi bentuk yang dihasilkan semakin tidak praktis. Sedangkan jika semakin banyak constraint yang digunakan, diperoleh penurunan tahanan gelombang yang lebih kecil, namun bentuk yang dihasilkan lebih baik.

Perlu dicatat bahwa bentuk-bentuk optimum di atas adalah optimum untuk sarat tertentu, kecepatan tertentu, dan kondisi constraint yang tertentu pula. Bentuk kapal yang optimum untuk suatu kondisi, belum tentu menjadi optimum jika kondisinya berubah. Juga perlu ditekankan bahwa pengertian optimum di dalam tugas akhir ini hanya ditinjau dari tahanan gelombang, sehingga banyak dijumpai bentuk lambung yang optimum untuk tahanan gelombang, tetapi mungkin tidak dapat diterima jika ditinjau dari sudut pandang yang lain.

Seberapa tingkat kebenaran dari hasil perhitungan tergantung dari sejauh mana formula integral Michell dapat diterima untuk perhitungan tahanan gelombang. Sebagaimana dijelaskan di muka, integral Michell didasarkan pada pendekatan thin ship theory dengan mengabaikan pengaruh viskositas fluida. Dari perhitungan didapatkan beberapa bentuk bulbous bow yang sangat besar dan bentuk lambung yang sangat lebar. Hal ini tentu saja bertentangan dengan asumsi thin ship theory. Sehingga, mungkin hasil perhitungan secara teoritis tidak sesuai dengan pengalaman praktis.

BAB VI
KESIMPULAN DAN SARAN

VI.1. KESIMPULAN

Dengan menggunakan teknik pemrograman kuadratik, persoalan optimisasi bentuk lambung kapal untuk tahanan gelombang minimum dapat diselesaikan. Walaupun constraints pertidaksamaan sebenarnya telah mencegah kenegatipan dari offsets lambung optimal, perilaku bentuk yang tidak teratur masih terjadi jika constraint yang ditentukan tidak cukup. Oleh sebab itu sangat penting untuk merumuskan sejumlah constraints yang sesuai untuk mendapatkan bentuk lambung kapal optimum yang praktis.

Dari analisa yang telah dilakukan, didapatkan kesimpulan tentang bentuk lambung optimum sebagai berikut :

- Bentuk optimum yang didapat dari perhitungan selalu mempunyai bulb atau lambung bergelombang.
- Disamping bulb haluan (bulbousbow), bulb juga bisa timbul di daerah midship yang letaknya cenderung lebih tinggi dari bulbousbow.
- Pada suatu kecepatan tertentu, jika dislasemen dinaikkan maka letak titik terlebar dari bulb akan semakin rendah.
- Jika displasemen dinaikkan, pada kecepatan rendah tambahan displasemen terdistribusi secara merata. Jika kecepatan dinaikkan tambahan displasemen akan semakin terkonsentrasi pada haluan

dan midship, sampai akhirnya pada kecepatan yang sangat tinggi tambahan displasemen terpusat di daerah haluan membentuk bulbousbow.

- Jika bulbousbow ditekan dengan menggunakan constraint, akan dihasilkan lambung bergelombang.

Prediksi teoritis mungkin tidak sepenuhnya sama dengan pengalaman praktis, terutama karena formula tahanan gelombang integral Michell didasarkan pada pendekatan thin ship theory. Tetapi, diharapkan bahwa analisa yang telah dilakukan bisa memberi beberapa gambaran untuk perancangan kapal optimum.

VI.2. SARAN

Perhitungan numerik di dalam tugas akhir ini didasarkan pada pendekatan lambung kapal dengan sejumlah grid yang dibentuk oleh 13 station dan 6 garis air. Perhitungan masih bisa dikembangkan dengan membuat grid yang lebih rapat, tetapi tentu saja dibatasi oleh kapasitas memory komputer.

Di dalam tugas akhir ini perhitungan dilakukan terhadap kapal series 60 block 60. Analisa perlu dilakukan terhadap beberapa model kapal lain pada kecepatan yang bervariasi serta pada kondisi constraints yang bervariasi secara sistematis.

Bentuk optimum hasil komputasi masih perlu dilakukan pengujian dengan menggunakan model, lebih-lebih karena analisa teoritis ini didasarkan pada asumsi kondisi permukaan bebas yang dilinierkan dan mengabaikan efek viscous dari fluida.

DAFTAR PUSTAKA

1. **Hsiung, Chi-Chao**, "Optimal Ship Forms for Minimum Wave Resistance", *Journal of Ship Research*, The Society of Naval Architects and Marine Engineers, Vol. 25, No. 2, 1981.
2. **Hsiung, Chi-Chao** dan **Shenyan, Dong**, "Optimal Ship Forms for Minimum Total Resistance", *Journal of Ship Research*, The Society of Naval Architects and Marine Engineers, Vol. 28, No. 3, 1984.
3. **Hsiung, Chi-Chao** dan **Shenyan, Dong**, "Applying Resistance Theory Used Quadratic Programmig Methode to Determine Optimal Ship Forms", *Workshop on Developments in Hull Form Design*, Maritime Research Institute Netherlands Wageningen, Vol. 2, Publication No. 785, 1985.
4. **Kyulevecheliev, S.** dan **Kovachev, S.**, "Interactive Hull Form Design for Minimum Wave Resistance", *Workshop on Developments in Hull Form Design*, Maritime Research Institute Netherlands Wageningen, Vol. 2, Publication No. 785, 1985.
5. **Philips, D.T** dan **Ravindran, A.**, *Operation Research: Principles and Practice*, John Wiley and Sons, New York.
6. **Arora, J.S.**, *Introduction to Optimum Design*, McGraw-Hill, New York, 1989.
7. **Lewis E.V.**, *Principles of Naval Architecture*, The Society of Naval Architects and Marine Engineers, New Jersey, Vol.2, 1988.

8. **Harvald, Sv. Aa**, *Resistance and Propulsion of Ships*, John Wiley and Sons, Canada, 1983.

9. **Lammeren, W.P.A., Troost, L,** dan **Konning, J.G.**, *Resistance Propulsion and Steering of Ships*, The Technical Publishing Company H. Stam Haarlem, Wageningen, 1948.

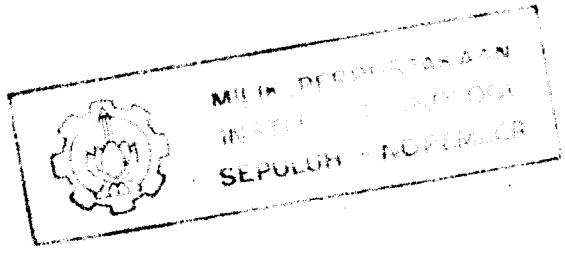
10. **Todd, F.H.**, *Series 60-Methodical Experiments with Models of Single-Screw Merchant Ship*, TMB Report No. 1712, 1963.

11. **Vossnack, E.**, "Historical Development in Hull Form Optimization", *Workshop on Developments in Hull Form Design*, Maritime Research Institute Netherlands Wageningen, Vol. 1, Publication No. 785, 1985.

12. **Fischer, O.**, *Borland Pascal / Turbo Pascal 7.0 Turbo Vision*, P.T. Dinastindo Adiperkasa Internasional, 1993.

13. **H.M., Jogiyanto**, *Teori dan Aplikasi Program Komputer Bahasa Pascal,* Penerbit Andi Offset Yogyakarta, Jilid I dan II, 1989.

14. **H.M., Jogiyanto**, *Pascal Tingkat Lanjutan*, Penerbit Andi Offset Yogyakarta, 1988



PENJELASAN PROGRAM

Program ditulis dalam bahasa Pascal, suatu bahasa program yang banyak dipakai secara luas untuk berbagai keperluan. Mengingat ukuran data yang diperlukan program sangat besar, maka program dicompile dengan menggunakan Borland Pascal 7.0, karena mendukung pemakaian mode protected yang memungkinkan untuk bekerja dengan menggunakan memory di atas 1 Mega byte. Penyusunan menu program menggunakan bahasa Turbo Vision 2.0 - Borland Pascal 7.0, dengan tujuan disamping agar diperoleh hasil yang memuaskan di dalam penyusunan program yang interaktif juga untuk menghindari membengkaknya ukuran program.

A.1. Struktur program

Program dibagi menjadi modul-modul yang terdiri atas program utama dan beberapa unit serta beberapa file pendukung. Pembagian program seperti ini dimaksudkan agar :

- Program menjadi lebih sistematis.
- Memudahkan di dalam pemahaman dan pelacakan program.
- Memudahkan di dalam pengembangan program di masa mendatang.

Adapun bagian-bagian yang terdapat di dalam program adalah :

- **Program SWR**

Program SWR adalah program untuk mendapatkan bentuk lambung yang optimum untuk tahanan gelombang minimum. Program ini merupakan bagian utama yang mengendalikan seluruh proses perhitungan. Dari sini data-data diolah dengan menggunakan prosedur-prosedur yang berada di unit-unit pendukung. Agar lebih interaktif, program disusun dalam bentuk menu dan dilengkapi dengan informasi bantuan (help) yang akan membimbing pemakai di dalam mengoperasikan program.

- **Unit WRESIST**

Seluruh proses yang berkaitan dengan perhitungan tahanan gelombang integral Michell dikerjakan di dalam unit ini.

- **Unit WQPRO**

Untuk memecahkan persoalan optimisasi bentuk lambung kapal, persoalan dirumuskan ke dalam bentuk persoalan program kuadratik standar. Harga dari konstanta-konstanta yang diperlukan untuk persoalan tersebut dihitung di dalam unit ini.

- **Unit WCPIVOT**

Di dalam unit ini, persoalan program kuadratik dipecahkan dengan metode Complementary Pivot. Dari unit ini didapatkan harga offsets lambung optimum.

- **Unit WCFORM**

Unit ini bertugas menghitung harga koefisien bentuk (C_w, C_b, C_m, C_p) dari kapal optimum.

- **Unit WLINES**

Seluruh proses yang berkaitan dengan visualisasi bentuk 2 dimensi lambung kapal dikerjakan di dalam unit ini.

- **Unit WPRINT**

Dengan menggunakan unit ini, input data dan output data dapat ditransfer ke alat pencetak.

- **Unit WTVC**

Untuk menghindari penulisan yang berulang, maka semua type, variabel, dan konstanta global yang digunakan didalam program utama dan semua unit pendukungnya, dideklarasikan di unit ini.

- **Unit WVAR**

Unit ini mendefinisikan type, variabel, dan konstanta global yang digunakan di dalam program utama.

- **Unit WCMDS**

Unit ini mendeklarasikan konstanta-konstanta untuk nomor perintah yang dipakai di dalam penyusunan menu program. Setiap nomor perintah akan menghubungkan suatu menu dengan suatu aksi tertentu.

- **Program SWRRESOURCE**

SWRRESOURCE adalah program untuk menghasilkan file resource yang diperlukan oleh program utama (SWR). Menu dan semua window dialog input output data disimpan di dalam file resource ini. Keuntungan dari adanya program ini adalah :

- Program utama (SWR) menjadi lebih sederhana.
- Jika ada perubahan di dalam menu atau window dialog, perbaikan cukup dilakukan di program ini tanpa harus merubah program utama (SWR).

□ **Teks bantuan SWR**

Teks bantuan SWR berisi text informasi bantuan (help) yang dipakai oleh program utama. Teks ini berada di dalam file SWR.TXT yang setelah dicompile dengan TVHC (Turbo Vision Help Compiler) akan menghasilkan file SWR.HLP dan unit WHELP.PAS.

□ **Unit WHELP**

Unit yang dihasilkan setelah kompilasi SWR.TXT ini berisi deklarasi konstanta-konstanta untuk nomor bantuan. Setiap nomor bantuan akan menghubungkan informasi bantuan di dalam file SWR.HLP dengan event yang sedang berlangsung.

A.2. Input dan output data

Data-data yang diperlukan program meliputi :

1. Ukuran utama kapal.
2. Titik-titik koordinat lambung kapal pembanding
2. Kecepatan kapal.
3. Constraints geometris lambung kapal yang bersifat pillihan.

Sedangkan yang menjadi output dari program adalah :

1. Gambar rencana garis kapal optimum.
2. Titik-titik koordinat lambung kapal optimum
3. Koefisien tahanan gelombang kapal.

A.3. Persyaratan hardware

Untuk dapat menjalankan program secara optimal, persyaratan perangkat keras (hardware) yang harus dipenuhi adalah :

- **Memory**

  Memory RAM yang bebas yang diperlukan oleh program adalah sebesar 1.75 Mb.

- **Disk**

  Selama proses perhitungan, data-data disimpan untuk sementara waktu di dalam disk. Memory disk yang diperlukan untuk penyimpanan ini adalah sebesar 400 Kb.

- **Monitor**

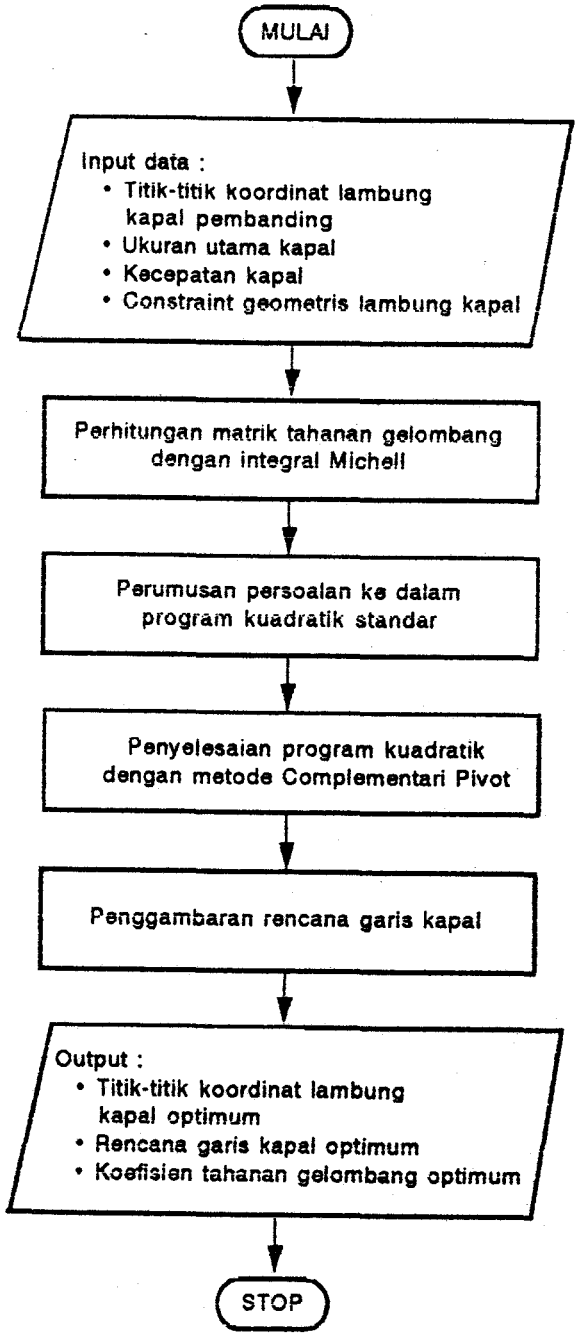
  Monitor yang diperlukan minimal CGA.

- **CPU**

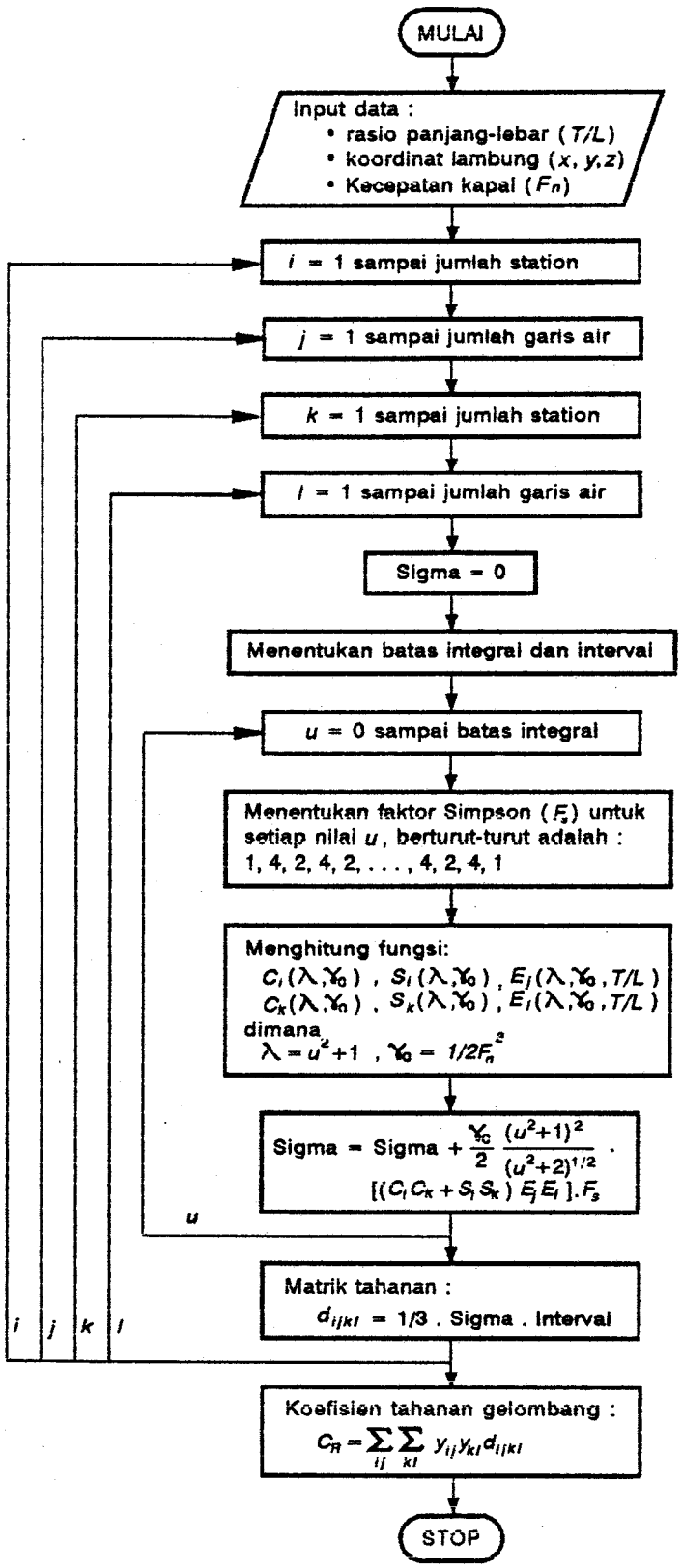
  Untuk dapat menjalankan program diperlukan processor yang mendukung pemakaian protected mode.

- **Mouse**

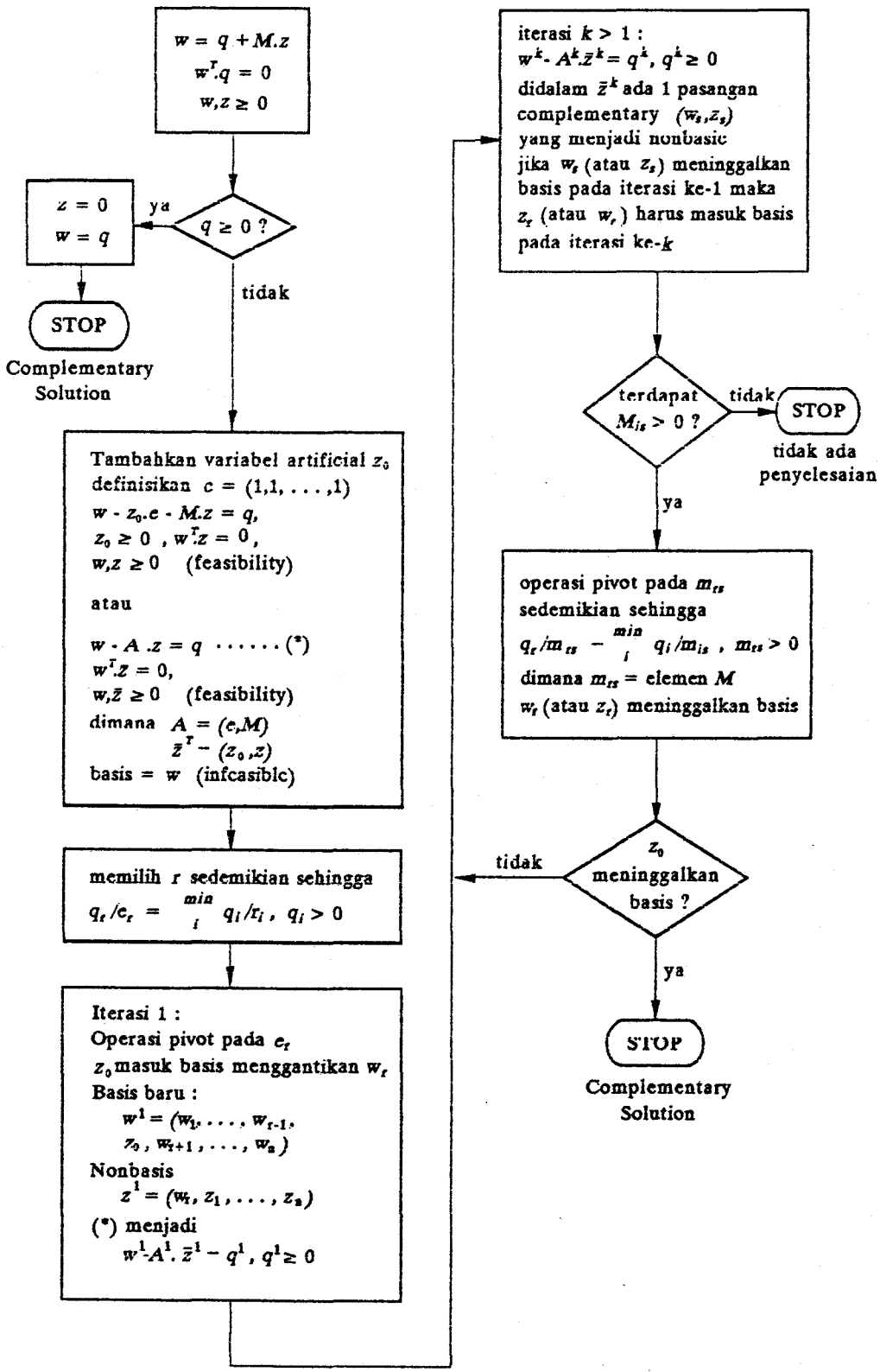
  Adanya mouse akan banyak membantu di dalam pengoperasian program.



Gambar B.1. Flowchart program utama



Gambar B.2. Flowchart Integral Michell



Gambar B.3. Flowchart Complementary Pivot

```
{************************************************************}
{*                                                          *}
{*    SWR Version 1.0                                       *}
{*    CopyRight (c) 1994 by Zulis Irawanto                  *}
{*    Teknik Perkapalan I T S                               *}
{*                                                          *}
{************************************************************}

Program SWR;

{  Program SWR adalah program untuk mendapatkan bentuk lambung          }
{  optimum untuk tahanan gelombang minimum                              }
{                                                                       }
{  Di samping menggunakan unit standard dari Pascal, program ini juga   }
{  menggunakan unit dari Turbo Vision 2.0 - Borland Pascal 7.0,         }
{  antara lain :                                                        }
{     Objects   App      Memory   Drivers   Views                      }
{     Menus     Dialogs  StdDlg   MsgBox    HelpFile                   }
{                                                                       }
{  File pendukung yang diperlukan program :                             }
{                                                                       }
{  ■  SWR.RES, yaitu file resource yang dihasilkan oleh program SWRRes  }
{     File ini berisi menu dan window dialog yang dipakai program SWR   }
{  ■  SERI60.HUL, berisi data koordinat lambung                         }
{     dari kapal series 60 block 60.                                    }
{  ■  WRESIST.MTR, berisi data matrik tahanan gelombang kapal Series    }
{     60 block 60 untuk data standar, yaitu T/L=0.053; Fn=0.289.        }
{  ■  SWR.HLP, berisi text bantuan (help) untuk program                 }

{$N+,E+,X+,S-,D-}
{$M 35000,0,655360}

uses
  WTVC, WNewType, WVar, WCmds, WHelp, WResist, WQPro,
  WCPivot, WLines, WCForm, WPrint,

  Dos, Crt, Objects, App, Memory, Drivers, Views, Menus,
  Dialogs, StdDlg, MsgBox, HelpFile;

type
  pHintStatusLine = ^tHintStatusLine;
  tHintStatusLine = Object (tStatusLine)
     function Hint(AHelpCtx: word): String; virtual;
  end;

  pSWR = ^tSWR;
  tSWR = object (TApplication)
          constructor Init;
          destructor Done; virtual;
          procedure GetEvent (var Event: TEvent); virtual;
```

```
          function GetPalette: PPalette; virtual;
          procedure HandleEvent (var Event: TEvent); virtual;
          procedure InitMenuBar; virtual;
          procedure InitStatusLine; virtual;
          procedure GetMainDimensions;
          procedure GetSpeed;
          procedure GetParrentOffsets;
          procedure GetWLAngle;
          procedure GetStAngle;
          procedure GetFixWLSt;
          procedure GetOfsBound;
          procedure GetFormCoef;
        end;
var
  ResFile     : TResourceFile;
  ParrentFile : PathStr;
  p: pDialog;
  w: word;

function tHintStatusLine.Hint (aHelpCtx: word): string;
var Info   : pStringList;
begin
  Info := pStringList (ResFile.Get('Help'));
  Hint := Info^.Get (aHelpCtx);
  Dispose (Info,Done);
end;

function CalcFileName (FileName:string): PathStr;
var
  EXEName: PathStr;
  Dir    : DirStr;
  Name   : NameStr;
  Ext    : ExtStr;
begin
  if Lo(DosVersion) >= 3 then EXEName := ParamStr(0)
  else EXEName := FSearch ('SWR.EXE', GetEnv('PATH'));
  FSplit (EXEName, Dir, Name, Ext);
  if Dir[Length(Dir)] = '\' then Dec(Dir[0]);
  CalcFileName := FSearch (FileName, Dir);
end;

constructor tSWR.Init;

var Event : tEvent;
    R     : tRect;
begin  { Init }
  MaxHeapSize := 8192;
  RegisterMenus;
  RegisterObjects;
  RegisterViews;
```

```
  RegisterApp;
  RegisterDialogs;
  RegisterStdDlg;
  RegisterHelpFile;
  RegisterType (RRealInputLine);
  RegisterType (rStringList);
  ResFile.Init (New (PBufStream, Init (CalcFileName ('SWR.RES'),
                stOpenRead, 2500)));
  Inherited Init;
  ParrentFile := CalcFileName ('SERI60.HUL');
  with MainDimS do
  begin
    Lpp := '100.0';
    B   := '13.33';
    T   := '5.300';
  end;
  SpeedS.nSpd   := '1';
  SpeedS.Spd[1] := '0.289';
  DisableCommands ([cmCResist, cmOfsOpt, cmCFormOpt, cmSaveDat,
                    cmPrintOutPut, cmGrShow, cmGrPrint]);
  Event.What    := evCommand;
  Event.Command := cmAbout;
  PutEvent (Event);
end;  { Init }

destructor tSWR.Done;
begin
  ResFile.Done;
  inherited Done;
end;

procedure tSWR.GetMainDimensions;
begin
  p := pDialog (ResFile.Get('MainDim'));
  w := ExecuteDialog (p,@MainDimS);
end;

Procedure tSWR.GetParrentOffsets;

procedure OpenFile;
begin
  if ExecuteDialog (PDialog(ResFile.Get('FileReadDlg')),
  @ParrentFile) <> cmCancel then
    if FSearch (ParrentFile, GetEnv('PATH'))='' then
    begin
      MessageBox (#13+^C'File tidak ditemukan !!!', nil,
                  mfError + mfOKButton);
      ParrentFile := CalcFileName ('SERI60.HUL');
    end;
end;
```

```
begin  { GetParrentOffsets }
  p := pDialog (ResFile.Get('OfsPar'));
  w := ExecuteDialog (p,@OfsPar);
  if w=cmOk then
  case OfsPar of
    0 : ParrentFile := CalcFileName ('SERI60.HUL');
    1 : OpenFile;
    2 : begin
          p := pDialog (ResFile.Get('XZCoord'));
          w := ExecuteDialog (p,@XZCoord);
          p := pDialog (ResFile.Get('YCoord'));
          w := ExecuteDialog (p,@YCoord);
        end;
  end;
end; { GetParrentOffsets }

procedure tSWR.GetSpeed;
begin
  p := pDialog (ResFile.Get('Speed'));
  w := ExecuteDialog (p,@SpeedS);
end;

procedure tSWR.GetFixWlSt;
begin
  p := pDialog (ResFile.Get('FixOfs'));
  w := ExecuteDialog (p,@Fix);
end;

procedure tSWR.GetFormCoef;
begin
  p := pDialog (ResFile.Get('CForm'));
  w := ExecuteDialog (p,@CForm);
end;

procedure tSWR.GetWLAngle;
begin
  p := pDialog (ResFile.Get('WLAngle'));
  w := ExecuteDialog (p,@WLAngleS);
end;

procedure tSWR.GetStAngle;
begin
  p := pDialog (ResFile.Get('StAngle'));
  w := ExecuteDialog (p,@StAngleS);
end;

procedure tSWR.GetOfsBound;
begin
  p := pDialog (ResFile.Get('OfsBound'));
  w := ExecuteDialog (p,@OfsBound);
end;
```

```
procedure tSWR.GetEvent (var Event: TEvent);

{  prosedur untuk menampilkan informasi (Help)  }
{     untuk event yang sedang berlangsung       }

var
  W        : PWindow;
  HFile    : PHelpFile;
  HelpStrm : PDosStream;

const
  HelpInUse     : Boolean = False;
  OnLineHlpInUse: Boolean = False;

begin
  inherited GetEvent(Event);
  if (Event.What=evCommand) and (Event.Command=cmHelp) then
  begin
    if (not HelpInUse) or (not OnLineHlpInUse) then
    begin
      HelpStrm := New(PDosStream, Init(CalcFileName
                    ('SWR.HLP'), stOpenRead));
      HFile := New(PHelpFile, Init(HelpStrm));
      if HelpStrm^.Status <> stOk then begin
        MessageBox ('File help tidak ditemukan !!!', nil,
                    mfError + mfOkButton);
        Dispose (HFile,Done);
      end
      else begin
        HelpInUse := true;
        if GetHelpCtx = hcOnLineHlp then OnLineHlpInUse := true;
        W := New(PHelpWindow, Init(HFile, GetHelpCtx));
        W^.HelpCtx := hcOnLineHlp;
        ExecView (W);
        Dispose (W,Done);
        HelpInUse := false;
        OnLineHlpInUse := false;
      end;
    end;
    ClearEvent (Event);
  end;
end;  { GetEvent }

function tSWR.GetPalette: PPalette;

const
  CNewColor = CAppColor + CHelpColor;
  P: string[Length(CNewColor)] = CNewColor;
begin
  GetPalette := @P[AppPalette];
end;
```

```
procedure tSWR.HandleEvent (var Event: TEvent);

{  prosedur untuk menjalankan aksi terhadap  }
{            pemilihan menu                  }

function power(I,J: word): word;
var
  Power1,K : word;
begin
  Power1:= 1;
  for K := 1 to J do power1 := Power1*I;
  Power := Power1;
end;

function SelectChBox(I,J:word): boolean;
begin
  if power(2,I-1) and J=power(2,I-1) then
  SelectChBox := true;
end;

procedure TransferData;

{ merubah data string ke data numerik }

var
  I, J, K, Err: word;
  F: text;
begin { TransferData }
  with MainDim do begin
    val (MainDimS.Lpp, Lpp, Err);
    val (MainDimS.B, B, Err);
    val (MainDimS.T, T, Err);
  end;

  With Speed do begin
    val (SpeedS.nSpd, nSpd, Err);
    for I:= 1 to nSpd do begin
      val (SpeedS.Spd[I], Spd[I], Err);
      val (SpeedS.Time[I], Time[I], Err)
    end
  end;

  with Coordinate do
  begin
    if (OfsPar=0) or (OfsPar=1) then begin
      Assign(F,ParrentFile);
      Reset(F);
      for I:= 1 to nSt do    Readln(F,X[I]);
      for J:= 1 to nWL do    Readln(F,Z[J]);
      for I:= 1 to nPoint do Readln(F,Y[I]);
      Close(F);
    end
```

```
    else begin
      for I:= 1 to nSt do    val(XZCoord.X[I], X[I], Err);
      for J:= 1 to nWL do    val(XZCoord.Z[J], Z[J], Err);
      for I:= 1 to nPoint do val(YCoord.Y[I], Y[I], Err);
    end;
  end;

  with Constraint do
  begin
    FixSt := [];
    for I:= 1 to nSt do
      if SelectChBox(I,Fix.St) = true then FixSt := FixSt+[I];
    FixWL := [];
    for I:= 1 to nWL do
      if SelectChBox(I,Fix.Wl) = true then FixWL := FixWL+[I];

    with StAngle do
    begin
      SelectSt := [];
      for I:= 1 to nSt do
      begin
        Val (StAngleS[I], Angle[I], Err);
        if Err=0 then SelectSt := SelectSt+[I]
      end;
    end;

    with WLAngle do
    begin
      SelectWL := [];
      for J:= 1 to nWL do
      begin
        Val (WLAngleS[J], Angle[J], Err);
        if Err=0 then SelectWL := SelectWL+[J]
      end;
    end;

    Cw.Select :=[]; Cb.Select :=[];
    Cm.Select :=[]; Cp.Select :=[];
    for I:= 1 to 2 do
    begin
      Val (CForm[1,I], Cw.Coef[I], Err);
      if Err=0 then Cw.Select := Cw.Select+[I];

      Val (CForm[2,I], Cb.Coef[I], Err);
      if Err=0 then Cb.Select := Cb.Select+[I];

      Val (CForm[3,I], Cm.Coef[I], Err);
      if Err=0 then Cm.Select := Cm.Select+[I];

      Val (CForm[4,I], Cp.Coef[I], Err);
      if Err=0 then Cp.Select := Cp.Select+[I];
    end;
```

```
    for I:= 1 to 13 do
    begin
      val (OfsBound.Bound1[I,1], MinOfs[1,I], Err);
      val (OfsBound.Bound2[I,1], MinOfs[2,I], Err);
      val (OfsBound.Bound1[I,2], MaxOfs[1,I], Err);

      if Err<>0 then MaxOfs[1,I] :=1E30;
      val (OfsBound.Bound2[I,2], MaxOfs[2,I], Err);
      if Err<>0 then MaxOfs[2,I] :=1E30;
    end;
  end;  { Constraint }
end;  { TransferData }

procedure ReplacePreviousData;

var
  I: byte;
begin
  with PrevData do
  begin
    with PrMainDim do
    begin
      Lpp:= MainDim.Lpp;
      B  := MainDim.B;
      T  := MainDim.T;
    end;

    with PrSpeed do
    begin
      nSpd := Speed.nSpd;
      for I:= 1 to nSpd do
      begin
        Spd[I] := Speed.Spd[I];
        Time[I]:= Speed.Time[I];
      end;
    end;

    with PrCoordinate do
    begin
      for I:= 1 to nSt do    X[I]:= Coordinate.X[I];
      for I:= 1 to nWL do    Z[I]:= Coordinate.Z[I];
      for I:= 1 to nPoint do Y[I]:= Coordinate.Y[I];
    end;
  end;  { PrevData }
end;  { ReplacePreviousData }

function DataEqualPrevData: boolean;

var
  Equal: boolean;
  I: byte;
```

```
begin
  Equal := true;
  with PrevData do
  begin
    with PrMainDim do
    begin
      if Lpp<> MainDim.Lpp then Equal := false;
      if B  <> MainDim.B   then Equal := false;
      if T  <> MainDim.T   then Equal := false;
    end;

    with PrSpeed do
    begin
      if nSpd <> Speed.nSpd then Equal := false;
      for I:= 1 to nSpd do
      begin
        if Spd[I] <> Speed.Spd[I]  then Equal := false;
        if Time[I]<> Speed.Time[I] then Equal := false;
      end;
    end;

    with PrCoordinate do
    begin
      for I:= 1 to nSt do    if X[I]<>Coordinate.X[I] then Equal:= false;
      for I:= 1 to nWL do    if Z[I]<>Coordinate.Z[I] then Equal:= false;
      for I:= 1 to nPoint do if Y[I]<>Coordinate.Y[I] then Equal:= false;
    end;
  end;  { PrevData }
  DataEqualPrevData := Equal;
end;  { DataEqualPrevData }

function Equal(S1,S2: single): boolean;
begin
  Equal := false;
  if Abs(S1-S2)<1E-6 then Equal := true
end;

function DefaultData: boolean;
begin
  DefaultData := true;
  if not Equal(MainDim.T/MainDim.Lpp,0.053) then DefaultData:= false;
  if not Equal(Speed.nSpd,1)                then DefaultData:= false;
  if not Equal(Speed.Spd[1],0.289)          then DefaultData:= false;
  if not Equal(OfsPar,0)                    then DefaultData:= false;
end;

procedure ReadResistanMatrix;

var
  I,J : byte;
  F   : text;
```

```
begin
  Assign(F, CalcFileName('WRESIST.MTR'));
  {$I-}
  Reset(F);
  {$I+}
  if IOResult=0 then
  begin
    for I:= 1 to nPoint do
      for J:= 1 to nPoint do
        Readln(F,D[I,J]);
    Close(F);
  end
  else  ResistanMatrix (MainDim.T, MainDim.Lpp, Speed,
                        Coordinate.X, Coordinate.Z, D)
end; { ReadResistanMatrix }

procedure Execution;

var
  I,J,K : byte;
  R     : tRect;
begin  { Execution }
  R.Assign(0, 0, 50, 12);
  p := New(PDialog, Init(R, 'Run'));
  with p^ do begin
    Options := Options or ofCentered;
    State := State or sfActive;
    Flags := Flags and not wfClose;
    R.Assign(2, 2, 25, 3);
    Insert (New(PLabel, Init(R, 'Proses perhitungan :',p)));
  end;
  Insert(p);

  TransferData;

  R.Assign(3, 4, 35, 5);
  p^.Insert(New(PStaticText,Init(R,'perhitungan matrik tahanan     :')));
  R.Assign(3, 5, 35, 6);
  p^.Insert(New(PStaticText,Init(R,'perumusan program kuadratik    :')));
  R.Assign(3, 6, 35, 7);
  p^.Insert(New(PStaticText,Init(R,'penyelesaian program kuadratik :')));

  R.Assign(37, 4, 47, 5);
  p^.Insert (New(PNewStaticText, Init(R, ' proses ')));
  if DefaultData then ReadResistanMatrix
  else if not DataEqualPrevData then
    ResistanMatrix (MainDim.T, MainDim.Lpp, Speed,
                    Coordinate.X, Coordinate.Z, D);
  p^.Insert (New(PStaticText, Init(R, ' selesai ')));

  ReplacePreviousData;
```

```
  R.Assign(37, 5, 47, 6);
  p^.Insert (New(PNewStaticText, Init(R, ' proses '))); Delay(1500);
  QuadraticProgram (Coordinate, MainDim, Constraint, D,
                    nModifPoint, ModifPoint, DeltaOfs);
  p^.Insert (New(PStaticText, Init(R, ' selesai ')));

  R.Assign(37, 6, 47, 7);
  p^.Insert (New(PNewStaticText, Init(R, ' proses '))); Delay(1500);
  ComplementaryPivot (Yo, SolType);
  p^.Insert (New(PStaticText, Init(R, ' selesai ')));

  R.Assign(2, 8, 48, 9);

  case SolType of
    1,2 : begin
            CResistance (D, Coordinate.Y, Cr0);
            for I:= 1 to nModifPoint do
               Coordinate.Y[ModifPoint[I]] :=
               Yo[I]+DeltaOfs[ModifPoint[I]];
            CResistance (D, Coordinate.Y, Cr1);

            p^.Insert(New(PNewStaticText, Init(R,
                          ' perhitungan berhasil')));
            HelpCtx := hcSol;
            EnableCommands ([cmCResist, cmOfsOpt, cmCFormOpt, cmSaveDat,
                             cmPrintOutPut, cmGrShow, cmGrPrint]);
          end;

    3   : begin
            p^.Insert(New(PNewStaticText, Init(R,
                          ' tidak ada penyelesaian')));
            HelpCtx := hcSol;
            DisableCommands ([cmCResist, cmOfsOpt, cmCFormOpt, cmSaveDat,
                              cmPrintOutPut, cmGrShow, cmGrPrint]);
          end;

    4   : begin
            p^.Insert(New(PNewStaticText, Init(R,
                      ' iterasi tidak berakhir proses dihentikan')));
            HelpCtx := hcSol;
            DisableCommands ([cmCResist, cmOfsOpt, cmCFormOpt, cmSaveDat,
                              cmPrintOutPut, cmGrShow, cmGrPrint]);
          end;
  end;  { case SolType }

  p^.Flags := p^.Flags or wfClose;
  p^.Redraw;
  ExecView(p);
  Dispose (p,Done);
  HelpCtx := hcNoContext;
end;  { Execution }
```

```
procedure LinesPlan (Destination: word);

var
  Err : word;
begin
  p := pDialog (ResFile.Get('Lines'));
  w := ExecuteDialog (p,@LinesData);
  if w<>cmOK then Exit;

  if Destination=2 then
  begin
    HelpCtx := hcPrintDlg;
    w := MessageBox (#13#3'Printer sudah siap ?', nil,
                     mfWarning+mfOKCancel);
    HelpCtx := hcNoContext;
    if w<>cmOK then Exit;
  end;

  with LinesData do
    LinesDrawing (MainDim, Coordinate, Lines, Section, Destination, Err);
  if Err<>0 then MessageBox (#13+^C'File BGI tidak ditemukan !!!',
                  nil, mfError + mfOKButton);
  ShowMouse;
  Redraw;
end;  { LinesPlan }

procedure ShowOptimalCForm;

var
  S1, S2, S3, S4: string [6];
begin
  CalcCForm (Coordinate, OptCw, OptCb, OptCm, OptCp);
  Str(OptCw:6:4,S1); Str(OptCb:6:4,S2);
  Str(OptCm:6:4,S3); Str(OptCp:6:4,S4);
  HelpCtx := hcCFormOptDlg;
  MessageBox(#13+^C'Cw :'+S1+'  Cb :'+S2+#13+^C'Cm :'+S3+'  Cp :'+S4+#13,
             nil, mfInformation or mfOKButton);
  HelpCtx := hcNoContext
end;

procedure ShowCResist;

var
  S1, S2: string [7];
begin
  Str(Cr0:7:5,S1); Str(Cr1:7:5,S2);
  HelpCtx := hcCResistDlg;
  MessageBox (#13+^C'Parrent Cr :'+S1+#13+^C'Optimal Cr :'+S2,
              nil, mfInformation + mfOKButton);
  HelpCtx := hcNoContext
end;
```

```
procedure ShowOptimalOffsets;

begin
  for w:= 1 to nPoint do
    Str(Coordinate.Y[w]:5:3, YOpt.Y[w]);
  p := pDialog (ResFile.Get('OfsOpt'));
  w := ExecuteDialog (p,@YOpt);
end;

procedure DoPrintOutput;

begin
  HelpCtx := hcPrintDlg;
  w := MessageBox (#13#3'Printer sudah siap ?', nil,
                   mfWarning + mfOKCancel);
  HelpCtx := hcNoContext;
  if w<>cmOK then Exit;
  PrintOutPut (Coordinate, MainDim, Speed, OptCw, OptCb,
               OptCm, OptCp, Cr1)
end;

procedure DoPrintInput;

begin
  TransferData;
  HelpCtx := hcPrintDlg;
  w := MessageBox (#13#3'Printer sudah siap ?', nil,
                   mfWarning + mfOKCancel);
  HelpCtx := hcNoContext;
  if w<>cmOK then Exit;
  PrintInPut (MainDim, Speed, Constraint);
end;

procedure SaveFile;

var OptName : PathStr;

procedure Save;

var I : byte;
    F : text;
begin
  Assign (F,OptName);
  Rewrite(F);
  with Coordinate do
  begin
    for I:= 1 to nSt do    Writeln(F,X[I]:3);
    for I:= 1 to nWL do    Writeln(F,Z[I]:3);
    for I:= 1 to nPoint do Writeln(F,Y[I]:3);
  end;
  Close(F);
end; { Save }
```

```
begin { SaveFile }
  OptName := '*.HUL';
  if ExecuteDialog (PDialog(ResFile.Get('FileSaveDlg')), @OptName)
  <> cmCancel then
  begin
    if FSearch (OptName, GetEnv('PATH'))<>''then
    begin
      w := MessageBox(#3'File '+OptName+#13#3'sudah ada. Ditindih ?',
                      nil, mfCancelButton + mfYesButton);
      if w=cmYes then Save;
    end
    else Save;
  end;
end; { SaveFile }

procedure DoAboutBox;
begin
  ExecuteDialog (PDialog(ResFile.Get('About')), nil);
end;

procedure ChangeDir;
begin
  ExecuteDialog (PDialog(ResFile.Get('ChDirDlg')), nil);
end;

procedure Quit;

begin  { Quit }
  if OfsPar=2 then
  begin
    HelpCtx := hcSaveBeforeQuit;
    w := MessageBox (^C'Data offsets parrent ship'+
         #13+#3'belum disimpan. Simpan ?',nil, mfYesNoCancel);
    if w=cmYes then SaveFile;
    if w=cmCancel then begin
      HelpCtx := hcNoContext;
      Exit;
    end;
  end
  else begin
    HelpCtx := hcExitDlg;
    w := MessageBox (#13+^C'Anda ingin keluar ?',nil,
                     mfYesButton + mfNoButton);
    if w<>cmYes then begin
      HelpCtx := hcNoContext;
      Exit
    end;
  end;
  Event.Command := cmQuit;
  PutEvent (Event);
end;  { Quit }
```

```
begin  { HandleEvent }
  Inherited HandleEvent (Event);
  if Event.What = evCommand then
    case Event.Command of
      cmAbout       : DoAboutBox;
      cmChDir       : ChangeDir;
      cmSpeed       : GetSpeed;
      cmMainDim     : GetMainDimensions;
      cmOfsPar      : GetParrentOffsets;
      cmFixOfs      : GetFixWlSt;
      cmCForm       : GetFormCoef;
      cmWLAngle     : GetWLAngle;
      cmStAngle     : GetStAngle;
      cmOfsBound    : GetOfsBound;
      cmPrintInput  : DoPrintInput;
      cmRun         : Execution;
      cmOfsOpt      : ShowOptimalOffsets;
      cmCResist     : ShowCResist;
      cmCFormOpt    : ShowOptimalCForm;
      cmSaveDat     : SaveFile;
      cmPrintOutPut: DoPrintOutPut;
      cmGrShow      : LinesPlan(1);
      cmGrPrint     : LinesPlan(2);
      cmExit        : Quit;
      else            Exit
    end;
  ClearEvent (Event);
end; { HandleEvent }

procedure tSWR.InitMenuBar;
begin
  MenuBar := PMenuBar (ResFile.Get('MenuBar'));
end;

procedure tSWR.InitStatusLine;

var
  R: tRect;
begin
  GetExtent(R);
  R.A.Y := R.B.Y-1;
  StatusLine := New(PHintStatusLine, Init(R,
    NewStatusDef(0, 0,
      NewStatusKey('~F1~ Help', kbF1, cmHelp,
      NewStatusKey('~F10~ Menu', kbF10, cmMenu,
      NewStatusKey('~Alt-X~ Exit', kbAltX, cmExit, nil))),

    NewStatusDef(1, 1,
      NewStatusKey('~F1~ Help', kbF1, cmHelp,
      NewStatusKey('~F5~ Zoom', kbF5, cmZoom,
      NewStatusKey('~Ctr+F5~ Move', kbCtrlF5, cmResize, nil))),
```

```
    NewStatusDef(2, 60,
      NewStatusKey('~F1~ Help', kbF1, cmHelp, nil),
    nil)))));
end;

var Application: tSWR;
begin
  Application.Init;
  Application.Run;
  Application.Done;
end.
```

```
{*************************************************************}
{*                                                           *}
{*    SWR Version 1.0                                        *}
{*    CopyRight (c) 1994 by Zulis Irawanto                   *}
{*    Teknik Perkapalan I T S                                *}
{*                                                           *}
{*************************************************************}

Unit WVar;

{$N+,E+}

{ unit ini mendeklarasikan variabel global yang dipakai program SWR }

Interface

uses WTVC;

type
  S7 = string [7];

var
  MainDim     : RMainDim;
  CoorDinate  : RCoord;
  Speed       : RSpeed;
  Constraint  : RConstraint;
  SolType     : byte;
  ModifPoint  : arr78B;
  nModifPoint : word;
  Yo          : Arr78;
  D           : Arr78_78;

  PrevData : record
    PrMainDim    : RMainDim;
    PrSpeed      : RSpeed;
    PrCoordinate : RCoord;
  end;

  MainDimS : record
    Lpp, B, T : S7
  end;

  SpeedS : record
    NSpd : string [1];
    Spd, time : array [1..6] of S7
  end;

  OfsPar : word;
```

```
  XZCoord : record
    Z : array [1..6] of S7;
    X : array [1..13] of S7
  end;

  YCoord : record
    Y : array [1..78] of S7
  end;

  YOpt: record
    Y : array[1..78] of S7
  end;

  Fix : record
    WL, St : word
  end;

  OfsBound : Record
    Bound1, Bound2 : array [1..13,1..2] of S7
  end;

  LinesData  : record
    Lines, Section : word
  end;

  CForm    : array [1..4,1..2] of S7;
  WLAngleS : array [1..6] of S7;
  StAngleS : array [1..13] of S7;
  Cr0, Cr1, OptCw, OptCb, OptCm, OptCp : single;
  DeltaOfs : arr78;

Implementation

END.
```

```
{*************************************************************}
{*                                                           *}
{*    SWR Version 1.0                                        *}
{*    CopyRight (c) 1994 by Zulis Irawanto                   *}
{*    Teknik Perkapalan I T S                                *}
{*                                                           *}
{*************************************************************}

Unit WTVC;

{$N+,E+}

{  unit ini berisi type, variabel, dan konstanta global  }
{       yang dipakai program utama dan semua unit        }

Interface

type
  Arr13    = array [1..13] of single;
  Arr6     = array [1..6] of single;
  Arr13_6  = array [1..13,1..6] of single;
  Arr78    = array [1..78] of single;
  Arr78_78 = array [1..78,1..78] of single;
  Arr78B   = array [1..78] of byte;

  RMainDim = record
               Lpp, B, T: single
             end;

  RCoord = record
             X : Arr13;
             Y : Arr78;
             Z : Arr6
           end;

  RSpeed = record
             nSpd      : byte;
             Spd, Time : Arr6
           end;

  RConstraint = Record
     FixWL : set of 1..6;
     FixSt : set of 1..13;
     Cw, Cb, Cm, Cp : Record
                 Coef     : array [1..2] of single;
                 Select   : set of 1..2;
               end;
```

```
    WLAngle : Record
                Angle    : arr6;
                SelectWL : set of 1..6;
              end;
    StAngle : Record
                Angle    : arr13;
                SelectSt : set of 1..13;
              end;
    MaxOfs, MinOfs : array [1..2] of Arr13;
  end;

   pA      = array [1..242] of ^Arr78;
   pQ      = ^Arr78_78;
   Arr242 = array [1..242] of single;

const
  nSt      = 13;
  Mid      = 7;
  nWL      = 6;
  nPoint   = 78;

Implementation

END.
```

```
{*************************************************************}
{*                                                           *}
{*    SWR Version 1.0                                        *}
{*    CopyRight (c) 1994 by Zulis Irawanto                   *}
{*    Teknik Perkapalan I T S                                *}
{*                                                           *}
{*************************************************************}

Unit WCmds;

{  Unit ini mendeklarasikan konstanta nomor perintah yang  }
{  dipakai oleh menu di dalam program SWR. Setiap nomor    }
{  perintah menghubungkan menu dengan aksi yang dilakukan  }

Interface

const
  cmAbout       =  101;
  cmMainDim     =  102;
  cmOfsPar      =  103;
  cmSpeed       =  104;
  cmFixOfs      =  105;
  cmOfsBound    =  106;
  cmCForm       =  107;
  cmWLAngle     =  108;
  cmStAngle     =  109;
  cmRun         =  110;
  cmOfsOpt      =  111;
  cmCResist     =  112;
  cmCFormOpt    =  113;
  cmSaveDat     =  114;
  cmGrShow      =  115;
  cmGrPrint     =  116;
  cmChDir       =  117;
  cmExit        =  118;
  cmPrintInput  =  119;
  cmPrintOutput =  120;

Implementation

END.
```

```
{*****************************************************************}
{*                                                               *}
{*    SWR Version 1.0                                            *}
{*    CopyRight (c) 1994 by Zulis Irawanto                       *}
{*    Teknik Perkapalan I T S                                    *}
{*                                                               *}
{*****************************************************************}

unit WHELP;

{  Unit ini mendeklarasikan konstanta untuk nomor bantuan    }
{  (Help). Setiap nomor bantuan menghubungkan event yang     }
{  sedang berlangsung dengan informasi yang diperlukan       }

interface

const

  hcAbout               = 8;
  hcCForm               = 20;
  hcCFormDlg            = 43;
  hcCFormOpt            = 30;
  hcCFormOptDlg         = 53;
  hcChDir               = 12;
  hcChDirDlg            = 50;
  hcConstr              = 17;
  hcCoordSys            = 48;
  hcCResist             = 29;
  hcCResistDlg          = 52;
  hcDosShel             = 10;
  hcExitDlg             = 51;
  hcFiles               = 9;
  hcFixOfs              = 18;
  hcFixOfsDlg           = 42;
  hcGraph               = 31;
  hcGrPrint             = 33;
  hcGrShow              = 32;
  hcHotKey              = 5;
  hcHowEnterDt          = 47;
  hcInput               = 13;
  hcLinesDlg            = 56;
  hcMainDim             = 14;
  hcMainDimDlg          = 36;
  hcMenuHotkey          = 3;
  hcNoContext           = 0;
  hcOfsBound            = 19;
  hcOfsBoundDlg         = 46;
  hcOfsOpt              = 28;
```

```
  hcOfsOptDlg             = 41;
  hcOfsPar                = 15;
  hcOfsParDlg             = 37;
  hcOnLineHlp             = 1;
  hcOutput                = 25;
  hcPrintDlg              = 54;
  hcPrintInput            = 23;
  hcPrintOutput           = 27;
  hcProgInfo              = 35;
  hcQuit                  = 11;
  hcRun                   = 24;
  hcSaveBeforeQuit        = 55;
  hcSaveDat               = 26;
  hcSaveOpenFileDlg       = 49;
  hcSol                   = 34;
  hcSpeed                 = 16;
  hcSpeedDlg              = 38;
  hcStAngle               = 22;
  hcStAngleDlg            = 45;
  hcSystem                = 7;
  hcUseMenuBar            = 4;
  hcUseOnLnHlp            = 2;
  hcUseProgram            = 6;
  hcWlAngle               = 21;
  hcWLAngleDlg            = 44;
  hcXZCoord               = 39;
  hcYCoord                = 40;

Implementation

END.
```

```
{*************************************************************}
{*                                                           *}
{*    SWR Version 1.0                                        *}
{*    CopyRight (c) 1994 by Zulis Irawanto                   *}
{*    Teknik Perkapalan I T S                                *}
{*                                                           *}
{*************************************************************}

Unit WNewType;

{$X+,S-,D-}

Interface

uses Objects, Dialogs, Views;

type
  PRealInputLine = ^TRealInputLine;
  TRealInputLine = object(TInputLine)
    function GetPalette: pPalette; virtual;
    function Valid(Command: Word): Boolean; virtual;
  end;

  PNewStaticText = ^TNewStaticText;
  TNewStaticText = Object (TStaticText)
      function GetPalette: pPalette; virtual;
  end;

procedure RegisterFields;

const
  RRealInputLine: TStreamRec = (
     ObjType: 10061;
     VmtLink: Ofs(TypeOf(TRealInputLine)^);
     Load:    @TRealInputLine.Load;
     Store:   @TRealInputLine.Store
  );

Implementation

uses  MsgBox;

procedure RegisterFields;
begin
  RegisterType(RRealInputLine);
end;
```

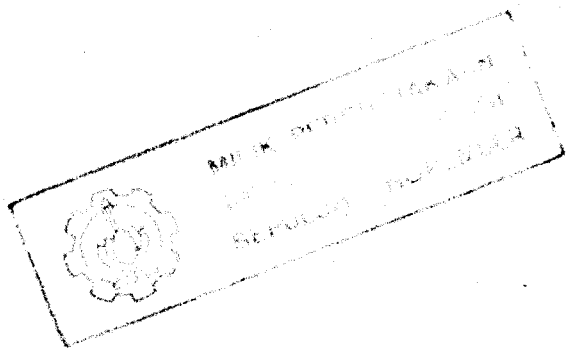
```
function TRealInputLine.Valid(Command: Word): Boolean;

var
  Ok: Boolean;
  Code: Integer;
  Value: Real;

begin
  Ok := True;
  if (Command <> cmCancel) and (Command <> cmValid) then
  begin
    Val(Data^, Value, Code);
    if (Data^<>'') and (Code <> 0) then
    begin
      Select;
      MessageBox (#13+^C'Invalid numeric format', nil,
                  mfError + mfOkButton);
      Ok := False;
    end;
  end;
  if Ok then
    Valid := TInputLine.Valid(Command)
  else
    Valid := False;
end;

function tRealInputLine.GetPalette: pPalette;
const
  p: string[3] = #24#24#24;
begin
  GetPalette := @p;
end;

function tNewStaticText.GetPalette: pPalette;
const
  p: String[1] = #32;
begin
  GetPalette := @p;
end;
```

```
{************************************************************}
{*                                                          *}
{*    SWR Version 1.0                                       *}
{*    CopyRight (c) 1994 by Zulis Irawanto                  *}
{*    Teknik Perkapalan I T S                               *}
{*                                                          *}
{************************************************************}

Unit WResist;

{$N+,E+,S-,D-}

Interface

uses WTVC;

procedure ResistanMatrix
          (var T, Lpp : single; var Speed : RSpeed;
          var X : Arr13; var Z : Arr6; var D : Arr78_78);

procedure CResistance (var D: Arr78_78; var Y: Arr78; var Cr: Single);

Implementation

procedure aResistanMatrix
          (var T, Lpp, Fn : single; var X : Arr13;
          var Z : Arr6; var D : arr78_78);

{*  prosedur perhitungan matrik tahanan  *}
{*      untuk satu kecepatan kapal       *}

var
  W, Lamda, Gama, U, Ci, Ck, Si, Sk : single;
  Ej, El, Sum :extended;
  I, J, K, L, M, N, Fs, Iu, Nu : word;
Const
  Step  = 0.03;
  Limit = 3.6;

function S (W: single; I: byte): single;
begin
  case I of
    1   : S := (cos(W*X[2])-cos(W*X[1]))
              /(W*(X[2]-X[1]));
    nSt : S := -(cos(W*X[nSt])-cos(W*X[nSt-1]))
              /(W*(X[nSt]-X[nSt-1]));
    else  S := (cos(W*X[I+1])-cos(W*X[I]))/
              (W*(X[I+1]-X[I]))-(cos(W*X[I])-
              cos(W*X[I-1]))/(W*(X[I]-X[I-1]))
  end
end;
```

```
function C (W: single; I: byte): single;

begin
  case I of
    1   : C := -(sin(W*X[2])-sin(W*X[1]))
                /(W*(X[2]-X[1]));
    nSt : C := (sin(W*X[nSt])-sin(W*X[nSt-1]))
               /(W*(X[nSt]-X[nSt-1]));
    else  C := -(sin(W*X[I+1])-sin(W*X[I]))/
               (W*(X[I+1]-X[I]))+(sin(W*X[I])-
               sin(W*X[I-1]))/(W*(X[I]-X[I-1]))
  end
end;

function E (W: single; J: byte): single;

begin
  case J of
    1   : E := (exp(W*(1-Z[2]))-exp(W*(1-Z[1])))/
               (sqr(W)*(Z[2]-Z[1]))+exp(W*(1-Z[1]))/W;
    nWL : E := -(exp(W*(1-Z[nWL]))-exp(W*(1-Z[nWL-1])))/
               (sqr(W)*(Z[nWL]-Z[nWL-1]))-exp(W*(1-Z[nWL]))/W;
    else  E := ((exp(W*(1-Z[J+1]))-exp(W*(1-Z[J])))/
               (Z[J+1]-Z[J])-(exp(W*(1-Z[J]))-exp
               (W*(1-Z[J-1])))/(Z[J]-Z[J-1]))/sqr(W)
  end
end;

Begin  { aResistanMatrix }

  Nu := round (Limit/Step);
  for I := 1 TO nPoint do
    for J := 1 TO nPoint do
      D[I,J] := 0;
  Gama := 1/(2*sqr(Fn));

  for I := 1 TO nSt do
    for J := 1 TO nWL do
    begin
      M := I+(J-1)*nSt;
      for K := 1 TO nSt do
        for L := 1 TO nWL do
        begin
          N := K+(L-1)*nSt;
          if N>=M then
          begin
            Sum := 0;
            for Iu := 0 TO Nu do
            begin
              U:=Step*Iu;
              Lamda:=sqr(U)+1;
```

```
            if (Iu=0) or (Iu=Nu) then
              Fs:=1
            else
            begin
              if Iu mod 2 = 0 then Fs:=2
              else Fs:=4
            end;

            W  :=2*Gama*Lamda;
            Ci := C(W,I);
            Ck := C(W,K);
            Si := S(W,I);
            Sk := S(W,K);

            W  := -2*sqr(Lamda)*Gama*T/LPP ;
            Ej := E(W,J);
            El := E(W,L);

            Sum := Sum+Fs*sqr(sqr(U)+1)/sqrT(sqr(U)+2)*
                   (Ci*Ck+Si*Sk)*Ej*El
          end;
          D[M,N]:=D[M,N]+(1/3*Gama/2*Step*Sum)
        end
      end
    end;
  for M := 1 TO nPoint do
    for N := 1 TO nPoint do
      if N<M then D[M,N] := D[N,M]
End; { aResistanMatrix }

procedure ResistanMatrix;

{*  prosedur perhitungan matrik tahanan  *}
{*     untuk beberapa kecepatan kapal    *}

type
  tD = array [1..78,1..78] of single;
var
  Sum, WitFac : single;
  I, J, K     : byte;
  pD          : ^tD;
Begin { ResistanMatrix }
  new(pD);
  for I := 1 to nPoint do
    for J := 1 to nPoint do
        pD^[I,J] := 0;

  with Speed do
  begin
    Sum := 0;
    for I := 1 to NSpd do
      Sum := Sum+Spd[I]*Time[I];
```

```
    for I := 1 to NSpd do
    begin
      if NSpd = 1 then WitFac := 1
      else WitFac := Spd[I]*Time[I]/Sum;
      aResistanMatrix (T, Lpp, Spd[I], X, Z, D);
      for J := 1 to nPoint do
        for K := 1 to nPoint do
              pD^[J,K] := pD^[J,K] + D[J,K]*WitFac
    end
  end;
  for J := 1 to nPoint do
    for K := 1 to nPoint do
          D[J,K] := pD^[J,K];
  Dispose(pD)

End;  { ResistanMatrix }

procedure CResistance;

{*  perhitungan koef.tahanan gelombang  *}

var I,J : byte;
Begin
  Cr := 0;
  for I := 1 to nPoint do
    for J := 1 to nPoint do
      Cr := Cr+D[I,J]*Y[I]*Y[J];
End;

END.
```



```
{************************************************************}
{*                                                          *}
{*    SWR Version 1.0                                       *}
{*    CopyRight (c) 1994 by Zulis Irawanto                  *}
{*    Teknik Perkapalan I T S                               *}
{*                                                          *}
{************************************************************}

Unit WQPro;

{$N+,E+,S-,D-}

Interface

Uses WTVC;

Procedure QuadraticProgram
          (var Coordinate: RCoord; var MainDim: RMainDim;
           var Constraint: RConstraint; var D: Arr78_78;
           var nModifPoint: word; var ModifPoint: Arr78B;
           var DeltaOfs: Arr78);

{  prosedur ini menghitung nilai konstanta [c], [H], [A],  }
{  dan [B] untuk persoalan program kuadratik :             }
{          minimumkan          : c'y + y'H y               }
{          dengan constraints : A y ≥ B                    }

Implementation

Procedure QuadraticProgram;

Var
  nConstr, I, J, J1, J2, K, L, M, N : byte;
  Max, Min, A1, A2, ACb, ACm        : single;
  FixPoint       : arr78B;
  SetModifPoint  : set of 1..78;
  A  : array [1..242] of ^Arr78;
  B  : array [1..242] of single;
  C  : Arr78;
  H  : ^Arr78_78;
  F  : Text;

Const
  GreatEq = 1;
  LessEq  = 2;

Begin  { QuadraticProgram }
  nConstr := 0;
  for I := 1 to 242 do New(A[I]);
  for I := 1 to 242 do
    for J := 1 to nPoint do A[I]^[J] := 0;
```

```
with Coordinate do
with Constraint do
begin { Constraint }
  nModifPoint := 0;
  SetModifPoint := [];
  for I := 1 to nSt do
    if not (I in fixSt) then
      for J := 1 to nWL do
        if not (J in fixWL) then
        begin
          nModifPoint := nModifPoint+1;
          K := I+(J-1)*nSt;;
          SetModifPoint := SetModifPoint + [K];
          ModifPoint[nModifPoint] := K
        end;
  K := 0;
  for I := 1 to nPoint do
    if Not (I in SetModifPoint) then
    begin
      Inc(K);
      FixPoint[K] := I
    end;

  { constraints batas atas offsets}

  for I := 1 to nSt do
    if not (I in FixSt) then
      for J := 1 to nWL do
        if not (J in FixWL) then
        begin
          K := I+(J-1)*nSt;
          if MaxOfs[1,I]<=MaxOfs[2,I]*Coordinate.Y[K] then
            Max := MaxOfs[1,I]
          else
            Max :=MaxOfs[2,I]*Coordinate.Y[K];
          if Max<1E20 then
          begin
            Inc(nConstr);
            B[nConstr]     := -Max;
            A[nConstr]^[K] := -1
          end;
        end;

  { Constraints koef. bidang garis air Cw }

  with Cw do
  if Not (nWL in FixWL) then
    for N := 1 to 2 do
      if N in Select then
      begin
        Inc(nConstr);
        B[nConstr] := 0;
```

```
      for I := 1 to nSt do
      begin
        case I of
          1   : A1 := (X[2]-X[1])/2;
          nSt : A1 := (X[nSt]-X[nSt-1])/2
          else  A1 := (X[I+1]-X[I-1])/2
        end;

        K := I+(nWL-1)*nSt;
        if K in SetModifPoint then
          A[nConstr]^[K] := A1
        else
          B[nConstr] := B[nConstr]-A1*Y[K]
      end;

      case N of
        GreatEq: B[nConstr] := B[nConstr]+Coef[N];
        LessEq :
          begin
            B[nConstr] := -B[nConstr]-Coef[N];
            for I := 1 to nModifPoint do
              A[nConstr]^[ModifPoint[I]] :=
              -A[nConstr]^[ModifPoint[I]]
          end;
      end;
    end;  { if N in Select }

{ constraints koefisien block Cb }

with Cb do
for N := 1 to 2 do
  if N in Select then
  begin
    Inc(nConstr);
    B[nConstr]:=0;

    for I := 1 to nSt do
    begin
      case I OF
        1   : A1 := (X[2]-X[1])/2;
        nSt : A1 := (X[nSt]-X[nSt-1])/2
        else  A1 := (X[I+1]-X[I-1])/2
      end;

      for J := 1 to nWL do
      begin
        case J OF
          1   : A2 := (Z[2]-Z[1])/2;
          nWL : A2 := (Z[nWL]-Z[nWL-1])/2
          else  A2 := (Z[J+1]-Z[J-1])/2
        end;
```

```
          K := I+(J-1)*nSt;
          if K in SetModifPoint then
            A[nConstr]^[K] := A1*A2
          else
            B[nConstr] := B[nConstr]-A1*A2*Y[K]
        end   { for J }
      end;  { for I }

      case N of
        GreatEq: B[nConstr] := B[nConstr]+Coef[N];
        LessEq : begin
                   B[nConstr] := -B[nConstr]-Coef[N];
                   for I := 1 to nModifPoint do
                     A[nConstr]^[ModifPoint[I]] :=
                     -A[nConstr]^[ModifPoint[I]]
                 end
      end
    end;   { if N in Select }

  { constraint koefisien midship Cm }

  with Cm do
  if Not (Mid in FixSt) then
    for N := 1 to 2 do
      if N in Select then
      begin
        Inc(nConstr);
        B[nConstr] := 0;
        for J := 1 to nWL do
        begin
          K := Mid+(J-1)*nSt;
          case J OF
            1     : A[nConstr]^[K] := (Z[2]-Z[1])/2;
            nWL   : A[nConstr]^[K] := (Z[nWL]-Z[nWL-1])/2;
            else    A[nConstr]^[K] := (Z[J+1]-Z[J-1])/2
          end;
          if Not (K in SetModifPoint) then
          begin
             B[nConstr] := B[nConstr]-A[nConstr]^[K]*Y[K];
             A[nConstr]^[K] := 0;
          end;
        end;
        case N of
          GreatEq: B[nConstr] := B[nConstr]+Coef[N];
          LessEq : begin
                     B[nConstr] := -B[nConstr]-Coef[N];
                     for I := 1 to nModifPoint do
                       A[nConstr]^[ModifPoint[I]] :=
                       -A[nConstr]^[ModifPoint[I]]
                   end
        end
      end;   { if N in Select }
```

```
{ constraint koef. prismatic Cp }

with Cp do
for N := 1 to 2 do
  if N in Select then
  begin
    Inc(nConstr);
    B[nConstr]:= 0;

    for I := 1 to nSt do
    begin
      case I OF
        1   : A1 := (X[2]-X[1])/2;
        nSt : A1 := (X[nSt]-X[nSt-1])/2
        else  A1 := (X[I+1]-X[I-1])/2
      end;
      ACm := 0;

      for J := 1 to nWL do
      begin
        case J OF
          1   : A2 := (Z[2]-Z[1])/2;
          nWL : A2 := (Z[nWL]-Z[nWL-1])/2
          else  A2 := (Z[J+1]-Z[J-1])/2
        end;

        ACb := A1*A2;
        if I=Mid then ACm := A2;
        K := I+(J-1)*nSt;

        if K in SetModifPoint then
          case N of
            GreatEq: A[nConstr]^[K] := ACb-ACm*Coef[N];
            LessEq : A[nConstr]^[K] := ACm*Coef[N]-ACb;
           end

        else
          case N of
            GreatEq: B[nConstr]:= B[nConstr]-(ACb-ACm*Coef[N])*Y[K];
            LessEq : B[nConstr]:= B[nConstr]+(ACb-ACm*Coef[N])*Y[K];
          end
      end
    end;
  end;

{ constraints sudut garis air }

with WLAngle do
for J := 1 to nWL do
  if J in (SelectWL-FixWL) then
    for I := 1 to nSt-1 do
```

```
      if [I,I+1]*FixSt <> [I,I+1] then
      begin
        Inc(nConstr);
        Angle[J] := Angle[J]/180* PI;
        K := I+(J-1)*nSt;
        B[nConstr] := (X[I]-X[I+1])*(2*MainDim.Lpp/MainDim.B)
                      *Sin(Angle[J])/Cos(Angle[J]);
        if I in FixSt then
          B[nConstr] := B[nConstr]-Y[K]
        else
          A[nConstr]^[K] := 1;

        if I+1 in FixSt then
          B[nConstr] := B[nConstr]+Y[K+1]
        else
          A[nConstr]^[K+1] := -1
      end;

{ constraint sudut garis station }

with StAngle do
for I := 1 to nSt do
  if I in (SelectSt-fixSt) then
  begin
    for J := 1 to nWL-1 do
      if [J,J+1]*FixWL <> [J,J+1] then
      begin
        Inc(nConstr);
        Angle[I] := Angle[I]/180* PI;
        K := I+(J-1)*nSt;
        B[nConstr] := (Z[J]-Z[J+1])*(2*MainDim.T/MainDim.B)
                      *Cos(Angle[I])/Sin(Angle[I]);
        if J in FixWL then
          B[nConstr] := B[nConstr]-Y[K]
        else
          A[nConstr]^[K] := 1;

        if J+1 in FixWL then
          B[nConstr] := B[nConstr]+Y[K+nSt]
        else
          A[nConstr]^[K+nSt] := -1
      end
  end;

{ matrix C }

for I := 1 to nModifPoint do
begin
  C[I] :=0;
  for J := 1 to nPoint-nModifPoint do
    C[I] := C[I]+2*D[ModifPoint[I],FixPoint[J]]*Y[FixPoint[J]]
end;
```

```
    { Hessian matrix [H] }

    New(H);
    for I := 1 to nPoint do
      for J := 1 to nPoint do H^[I,J] := 0;
    for I := 1 to nModifPoint do
      for J := 1 to nModifPoint do
        H^[I,J] := D[ModifPoint[I],ModifPoint[J]];

    { matrix [B] & [C] untuk constraints batas bawah offsets }

    for I := 1 to nSt do
      if not (I in FixSt) then
        for J := 1 to nWL do
          if not (J in FixWL) then
          begin
            K := I+(J-1)*nSt;
            if MinOfs[1,I]>MinOfs[2,I]*Coordinate.Y[K] then
               DeltaOfs[K] := MinOfs[1,I]
            else
               DeltaOfs[K] := MinOfs[2,I]*Coordinate.Y[K]
          end;

    if nConstr <> 0 then
      for I := 1 to nConstr do
        for J := 1 to nModifPoint do
          B[I] := B[I]-A[I]^[ModifPoint[J]]*DeltaOfs[ModifPoint[J]];
    for I := 1 to nModifPoint do
      for J := 1 to nModifPoint do
        C[I] := C[I]+2*H^[I,J]*DeltaOfs[ModifPoint[J]];

  end;  {Constraint}

  Assign(F,'WQPRO.DAT');
  Rewrite(F);

  Writeln(F,nModifPoint:2);
  Writeln(F,nConstr:3);

  for I := 1 to nModifPoint do
    Writeln(F,C[I]:13);
  for I := 1 to nConstr do
    Writeln(F,B[I]:13);

  for I := 1 to nModifPoint do
    for J := 1 to nModifPoint do
      Writeln(F,H^[I,J]:13);

  for I := 1 to nConstr do
    for J := 1 to nModifPoint do
      Writeln(F,A[I]^[ModifPoint[J]]:13);
```

```
  Close(F);
  Dispose(H);
  for I := 1 to 242 do
    Dispose(A[I]);

End; { QuadraticProgram }

END.
```

```
{*************************************************************}
{*                                                           *}
{*    SWR Version 1.0                                        *}
{*    CopyRight (c) 1994 by Zulis Irawanto                   *}
{*    Teknik Perkapalan I T S                                *}
{*                                                           *}
{*************************************************************}

Unit WCPivot;

{$N+,E+,S-,D-}

Interface

Uses WTVC;

procedure ComplementaryPivot (var Y: Arr78; var SolType: byte);

{  prosedur ini mencari nilai optimum [y] untuk persoalan    }
{  program kuadratik :                                       }
{          minimumkan          : c'y + y'H y                 }
{          dengan constraints : A y ≥ B                      }
{                                                            }
{  data konstanta [c], [H], [A], [B], jumlah variabel, dan   }
{  jumlah constraints dibaca dari file WQPro.Dat yang        }
{  dihasilkan oleh unit WQPro                                }

Implementation

procedure ComplementaryPivot;

type
  tM = Array[1..641] of double;

var
  A, B, C, H : single;
  nVar, nConstr, nRow, nCol, Row, Col : word;
  I, J, K, Zo, EnterBasic, LeftBasic  : word;
  M     : array [1..320] of ^tM;
  q     : Array [1..320] of double;
  Basic : array [1..320] of word;
  F     : Text;
  Pivot, Mcol, Ratio, Min  : double;
  NonNegativQ, ZoNotInBasis, NoPivotFound : boolean;

const
  Complementary         = 1;
  AlmostComplementary   = 2;
  Ray                   = 3;
  Degenerate            = 4;
```

```
Begin
  Assign(F,'WQPRO.DAT');
  Reset(F);

  Readln(F,nVar);
  Readln(F,nConstr);
  nRow := nVar+nConstr;
  nCol := 2*nRow+1;

  { nilai awal [q] }

  for I := 1 to nVar do
    begin
      Readln(F,C);
      q[I] := C
    end;
  for I := 1 to nConstr do
    begin
      Readln(F,B);
      q[I+nVar] := -B
    end;

  { almost complementary Solution }

  NonNegativQ := true;
  for I := 1 to nRow do
    if q[I] < 0 then NonNegativQ := false;
  if NonNegativQ then
  begin
    for I := 1 to nVar do
      Y[I] := q[I];
    Close(F);
    Erase(F);
    SolType := AlmostComplementary;
    EXIT
  end;

  { Initial value of M }

  for I := 1 to nRow do
  begin
    new(M[I]);
    for J := 1 to nCol do
      M[I]^[J] :=0
  end;

  Zo := nCol;
  for I := 1 to nRow do
  begin
    M[I]^[I] := 1;
    M[I]^[Zo] := -1
  end;
```

```
for I := 1 to nVar do
  for J := 1 to nVar do
    begin
      Readln(F,H);
      M[I]^[J+nRow] := -2*H
    end;

if nConstr>0 then
begin
for I := 1 to nConstr do
  for J := 1 to nVar do
    begin
      Readln(F,A);
      M[I+nVar]^[J+nRow] := -A
    end;
for I := 1 to nVar do
  for J := 1 to nConstr do
      M[I]^[J+nRow+nVar] := -M[J+nVar]^[I+nRow];
end;
Close(F);
Erase(F);

{ variabel basis awal }

for I := 1 to nRow do
  Basic[I] := I;

{ Iterasi 1 }

Min := 0;
for I := 1 to nRow do
  if q[I] < Min then
  begin
    Min := q[I];
    Row := I
  end;
EnterBasic := Zo;
LeftBasic := Basic[Row];
Basic[Row] := EnterBasic;

{  operasi pivot  }

for I := 1 to nRow do
  if I <> Row then
  begin
    q[I] := q[I]-q[Row];
    for J := 1 to nCol do
      M[I]^[J] := M[I]^[J]-M[row]^[J]
  end;
q[Row] := -q[Row];
for J := 1 to nCol do
  M[row]^[J] := -M[row]^[J];
```

```
for I := 1 to nRow do
  for J := 1 to nCol do
    if abs(M[I]^[J]) < 1E-80 then M[I]^[J] := 0;

{ Iterasi berikutnya }

ZoNotInBasis := false;
Repeat

  { degenerate solution }
  for I := 1 to nRow do
    if Abs(q[I])< 1E-80 then
    begin
      SolType := Degenerate;
      for J := 1 to nRow do
        Dispose(M[J]);
      Exit
    end;

  if LeftBasic <= nRow then
    EnterBasic := LeftBasic+nRow
  else
    EnterBasic := LeftBasic-nVar-nConstr;
  Col := EnterBasic;

  NoPivotFound := true;
  for I := 1 to nRow do
    if M[I]^[col]>0 then NoPivotFound := false;

  { ray solution }

  if NoPivotFound then
  begin
    SolType := Ray;
    for I := 1 to nRow do
      Dispose(M[I]);
    Exit
  end;

  { memilih baris pivot }

  Min := 1E300;
  for I := 1 to nRow do
    if M[I]^[col]>0 then
    begin
      Ratio := q[I]/M[I]^[col];
      if Ratio<Min then
      begin
        Min := Ratio;
        Row := I
      end
    end;
```

```
    if Basic[Row]=Zo then
      ZoNotInBasis := true;
    Pivot := M[row]^[col];
    LeftBasic := Basic[Row];
    Basic[Row] := EnterBasic;

    { operasi pivot }

    for I := 1 to nRow do
      if I<>Row then
      begin
        Mcol := M[I]^[col];
        q[I] := q[I]-q[Row]/Pivot*Mcol;
        for J := 1 to nCol do
          M[I]^[J] := M[I]^[J]-M[row]^[J]*Mcol/Pivot
      end;

    q[Row] := q[Row]/Pivot;
    for J := 1 to nCol do
      M[row]^[J] := M[row]^[J]/Pivot;

    for I := 1 to nRow do
      for J := 1 to nCol do
        if abs(M[I]^[J]) < 1E-80 then
          M[I]^[J]:= 0;
  until ZoNotInBasis;  {akhir iterasi}

  for I := 1 to nRow do
    Dispose(M[I]);

  { Complementary solution }

  SolType := Complementary;
  for I := 1 to nVar do
      Y[I] := 0;

  for I := 1 to nRow do
    if (Basic[I] > nRow) and (Basic[I] <= nRow+nVar) then
      Y[Basic[I]-nVar-nConstr] := q[I];
End;

END.
```

```
{************************************************************}
{*                                                          *}
{*    SWR Version 1.0                                       *}
{*    CopyRight (c) 1994 by Zulis Irawanto                  *}
{*    Teknik Perkapalan I T S                               *}
{*                                                          *}
{************************************************************}

Unit WLines;

{$N+,E+,S-,D-}

Interface

uses WTVC;

procedure  LinesDrawing (MainDim: RMainDim; var Coordinate: RCoord;
                         Lines, Section, Destination: word;
                         var Err: word);

Implementation

uses Crt, Dos, Graph;

type
  Arr13_2  = array [1..13,1..2]  of single;
  Arr150_2 = array [1..600,1..2] of single;

procedure  CubicSpline (N: byte; Step: single; var Input: Arr13_2;
                        var Output: Arr150_2);

{  prosedur perhitungan interpolasi cubicspline dari [N] variabel  }
{      [Input] dengan interval [step] menghasilkan [Output]        }

var
  M : array [1..13,1..13] of single;
  G : array [1..4,1..2] of single;
  R, Tangen : array [1..13,1..2] of single;
  F         : array [1..4] of single;
  Point     : array [1..2] of single;
  I, J, K, L, Icount : byte;
  Pv, A, Coef, Tau   : single;

Begin  { CubicSpline }

  for I := 1 TO N do
    for J := 1 TO N do
      M[I,J] := 0;
```

```
for I := 2 to N-1 do
begin
  M[I,I-1] := 1;
  M[I,I] := 4;
  M[I,I+1] := 1
end;

for I := 2 to N-1 do
  for J := 1 to 2 do
    R[I,J] := 3*(Input[I+1,J]-Input[I-1,J]);

M[1,1] := 1;
M[1,2] := 0.5;
M[N,N-1] := 2;
M[N,N] := 4;

for J := 1 to 2 do
begin
  R[1,J] := 3/2*(Input[2,J]-Input[1,J]);
  R[N,J] := 6*(Input[N,J]-Input[N-1,J])
end;

for I := 1 to N do
begin
  Pv := M[I,I];
  M[I,I] :=1;
  for J := 1 to N do
    M[I,J] := M[I,J]/Pv;

  for K := 1 to N do
    if K <> I then
    begin
      A := M[K,I];
      M[K,I] := 0;
      for J := 1 to N do
        M[K,J] := M[K,J]-A*M[I,J]
    end
end;

for I := 1 to N do
  for J := 1 to 2 do
  begin
    Tangen[I,J] := 0;
    for K := 1 to N do
      Tangen[I,J] := Tangen[I,J] + M[I,K] * R[K,J]
  end;

Icount := 0;
for I := 2 to N do
begin
  for J := 1 to 2 do
```

```
    begin  { for J }
      G[1,J] :=  Input[I-1,J];
      G[2,J] :=  Input[I,J];
      G[3,J] :=  Tangen[I-1,J];
      G[4,J] :=  Tangen[I,J]
    end;

    for J := 0 to round (1/Step-1) do
    begin
      Icount := Icount+1;
      Tau := J * Step;

      F[1] := Tau*(2*sqr(Tau)-3*Tau)+1;
      F[2] := Tau*(-2*sqr(Tau)+3*Tau);
      F[3] := Tau*(sqr(Tau)-2*Tau+1);
      F[4] := Tau*(sqr(Tau)-Tau);

      for K := 1 to 2 do
      begin
        Point[K] := 0;
        for L := 1 to 4 do
          Point[K] := Point[K] + F[L] * G[L,K]
      end;
      for K := 1 to 2 do
        Output[Icount,K] := Point[K]
    end
  end;
  Icount := Icount+1;
  for J := 1 to 2 do
    Output[Icount,J] := Input[N,J]
End;  { CubicSpline }

procedure PrintGraph;

const
  Esc = 27;
  Linefeed = 10;
  Bits : array[0..7] of byte = (128,64,32,16,8,4,2,1);

procedure GiveByte(b: byte);
var
  Regs: registers;
begin
  with Regs do
  begin
    AH := 0;
    AL := b;
    DX := 0;
    intr($17,Regs);
  end;
end;
```

```
procedure Space (spc: byte);
begin
  GiveByte(Esc);
  GiveByte(ord('3'));
  GiveByte(spc);
end;

Procedure SetGraph;
begin
  GiveByte(Esc);
  GiveByte(ord('L'));
  GiveByte(128);
  GiveByte(2);
end;

var
  Lenght, Height, line, X, Y, Top, Btm, pos : integer;
  bytes : byte;
  bytess : array[0..639] of byte;

begin
  Lenght := GetMaxX;
  Height := GetMaxY;
  Top:=0;
  Btm:= 7;
  Pos := 0;
  bytes := 0;
  {$B-}

  for line := 0 to (Height div 8) do
  begin
    Space(1);
    SetGraph;
    for X:=0 to Lenght do
    begin
      for Y:= Top to Btm do
      begin
        if getpixel(X,Y) <> 0 then
          inc(bytes,bits[Pos]);
        inc(Pos);
      end;

      bytess[X] := bytes;
      GiveByte(bytes);
      bytes := 0;
      Pos := 0;
    end;

    GiveByte(Linefeed);
    Space(22);
    SetGraph;
```

```
    for X:=0 to Lenght do
      GiveByte(bytess[X]);
    GiveByte(Linefeed);
    inc(Top,8);
    inc(Btm,8);
  end;
  {$B+}
End;  { PrintGraph }

function CalcBGIDir: PathStr;

{ mencari directory letak file BGI }

var
  EXEName: PathStr;
  Dir: DirStr;
  Name: NameStr;
  Ext: ExtStr;

begin
  if Lo(DosVersion) >= 3 then EXEName := ParamStr(0)
  else EXEName := FSearch('WH.EXE', GetEnv('PATH'));
  FSplit(EXEName, Dir, Name, Ext);
  if Dir[Length(Dir)] = '\' then Dec(Dir[0]);
  CalcBGIDir := Dir;
end;

procedure  LinesDrawing;

{  prosedur penggambaran rencana garis kapal  }

Type
  tPolygon = array [1..600] of Pointtype;
var
  Driver, Mode, I, J, K, Npoint : Integer;
  BaseL, CenterL, AP, FP, MidSect, L, B, T, Xasp, Yasp,
  Xa, Ya, Xb, Yb, Sectist, SectEnd : Word;
  Step      : Single;
  iStr, Pos : string[5];
  Input     : Arr13_2;
  Output    : Arr150_2;
  pPolygon  : ^tPolygon;

Begin { LinesDrawing }
  Driver := detect;
  InitGraph(Driver, Mode, CalcBGIDir);
  Err := GraphResult;
  if Err <> grOk then exit;
  SetBkColor(0);
  SetColor(15);
  GetAspectRatio(Xasp, Yasp);
```

```
with Coordinate do
begin
  if Lines=0 then begin

    {* BODY PLAN *}

    T := Round(0.8*GetMaxY);
    B := Round(Yasp/Xasp*T*MainDim.B/MainDim.T);
    if B div 2 > Round(0.7*GetMaxX) then
    begin
      B := Round(1.4*GetMaxX);
      T := Round(Xasp/Yasp*B*MainDim.T/MainDim.B);
    end;

    Sect1St:=1;
    SectEnd:=nSt;
    Case Section of
      1: begin
           BaseL   := Round(0.85*GetMaxY);
           CenterL := Round(0.1*GetMaxX);
           SectEnd := Mid;
         end;
      2: begin
           BaseL   := Round(0.85*GetMaxY);
           CenterL := Round(0.1*GetMaxX);
           Sect1St := Mid;
         end;
      0: begin
           BaseL   := Round(0.6*GetMaxY);
           CenterL := Round(0.5*GetMaxX);
           T       := T div 2;
           B       := B div 2;
         end;
    end;

    { garis air }

    if Section=0 then  Xa:= CenterL - B div 2
    else Xa:= CenterL;
    Xb:= CenterL + B div 2;

    for I := 1 to nWL do
    begin
      Ya := BaseL - Round(T*Z[I]);
      Yb := Ya;
      Line (Xa,Ya,Xb,Yb);
      SetTextJustify (1,1);
      Str (Z[I]:5:3, iStr);
      OutTextXY (Xa-40, Ya, iStr);
      OutTextXY (Xb+40, Ya, iStr)
    end;
```

```
{ buttock line }

Ya := BaseL;
Yb := Ya-T;
for I := 0 TO 4 do
begin
  if Section=0 then
       Xa := CenterL+Round((I-2)*B/4)
  else Xa := CenterL+Round(I*B/8);
  Xb := Xa;
  Line (Xa, Ya, Xb, Yb);

  SetTextJustify (1,1);
  if Section=3 then
       Str ((Abs(I-2)/4):3:2, iStr)
  else Str ((I/4):3:2, iStr);
  OutTextXY (Xa, Ya+10, iStr);
  OutTextXY (Xa, Ya+10, iStr)
end;

{ section line }

Pos := 'Fore';
for I := Sectist to SectEnd do
begin
  for J := 1 to nWL do
  begin
    K := I+(J-1)*nSt;
    Input[J,1] := Y[K];
    Input[J,2] := Z[J]
  end;

  Step := 0.05;
  CubicSpline (nWL, Step, Input, Output);
  Npoint :=  round ((nWL-1)/Step) + 1;
  New(pPolygon);

  for J := 1 to nPoint do
  begin
    if (Section=0) and (Pos<>'Fore') then
      pPolygon^[J].X :=  CenterL-Round(Output[J,1]*B/2)
    else
      pPolygon^[J].X :=  CenterL+Round(Output[J,1]*B/2);
    pPolygon^[J].Y :=  BaseL-Round(Output[J,2]*T)
  end;

  if (Section=0) and (I=Mid) and (Pos='Fore') then
  begin
    I   := Mid-1;
    Pos := 'Aft'
  end;
```

```
    drawPoly (Npoint, pPolygon^);
    Dispose (pPolygon);
  end;  { for I }
end  { Lines=0 }

else  { I<>0 }
begin

  {* WATER PLAN *}

  L  := Round(1.8*GetMaxX);
  B  := Round(Xasp/Yasp*L*MainDim.B/MainDim.Lpp);

  {     FORE     }

  if Section in [0,1] then
  begin
  MidSect := 45;
  FP := MidSect + L div 2;
  case Section of
    1: CenterL := Round(0.55*GetMaxY);
    0: CenterL := Round(0.45*GetMaxY);
  end;
  { buttock line }

  Xa := FP;
  Xb := MidSect;
  for I := 0 TO 4 do
  begin
    Ya := CenterL-Round(I*B/8);
    Yb := Ya;
    Line (Xa, Ya, Xb, Yb);
    SetTextJustify (LeftText,1);
    Str (I/4:4:2, iStr);
    OutTextXY (0, Ya, iStr);
  end;

  { section line }

  Ya := CenterL;
  Yb := Ya-Round(B div 2);
  for I := 1 to Mid do
  begin
    Xa := FP-Round(X[I]*L);
    Xb := Xa;
    Line (Xa, Ya, Xb, Yb);
    SetTextJustify (1,1);
    Str(I,iStr);
    OutTextXY (Xa, Ya+10, iStr);
    if I=1 then OutTextXY (Xa, Ya+25, 'FP');
  end;
```

```
  { garis air }

  for I := 1 TO nWL do
  begin
    for J := 1 TO Mid+1 do
    begin
      K := J+(I-1)*nSt;
      Input[J,1] := X[J];
      Input[J,2] := Y[K]
    end;

    Step := 0.05;
    CubicSpline (Mid+1, Step, Input, Output);
    nPoint := round((Mid-1)/Step)+1;
    New (pPolygon);

    for J := 1 TO nPoint do
    begin
      pPolygon^[J].X :=  FP-Round(Output[J,1]*L);
      pPolygon^[J].Y :=  CenterL-Round(Output[J,2]*B/2);
    end;

    DrawPoly (nPoint, pPolygon^);
    Dispose (pPolygon);
  end;
end;   { Section in [0,1] }

  {     AFTER     }

if Section in [0,2] then
begin
  AP := 45;
  MidSect := AP + L div 2;
  case Section of
    2: CenterL := Round(0.55*GetMaxY);
    0: CenterL := Round(0.90*GetMaxY);
  end;

  { buttock line }

  Xa := AP;
  Xb := MidSect;
  for I := 0 TO 4 do
  begin
    Ya := CenterL-Round(B*I/8);
    Yb := Ya;
    Line (Xa, Ya, Xb, Yb);
    SetTextJustify (LeftText,1);
    Str (I/4:4:2, iStr);
    OutTextXY (0, Ya, iStr);
  end;
```

```
        { section line }

        Ya := CenterL;
        Yb := Ya - B div 2;
        for I := Mid to nSt do
        begin
          Xa := MidSect-Round(L*X[I])+L div 2;
          Xb := Xa;
          Line (Xa, Ya, Xb, Yb);
          SetTextJustify (1,1);
          Str(I,iStr);
          OutTextXY (Xa, Ya+10, iStr);
          if I=nSt then OutTextXY (Xa, Ya+25, 'AP')
        end;

        { garis air }

        for I := 1 to nWL do
        begin
          for J := Mid-1 to nSt do
          begin
            K := J+(I-1)*nSt;
            Input[J-5,1] := X[J];
            Input[J-5,2] := Y[K]
          end;

          Step := 0.05;
          CubicSpline (Mid+1, Step, Input, Output);
          nPoint := round((Mid-1)/Step)+1;
          New (pPolygon);

          for J := 1 to nPoint do
          begin
            pPolygon^[J].X :=  MidSect-Round(L*Output[Round
                               (1/step+j),1]) + L div 2;
            pPolygon^[J].Y :=  CenterL-round(Output[J+Round
                               (1/step),2]*B/2)
          end;
          DrawPoly (nPoint, pPolygon^);
          Dispose (pPolygon);
        end; { for I }
      end; { if Section in [0,2] }
    end; { Lines<>0 }
  end; { Coordinate }

  if Destination=2 then PrintGraph;
  repeat until KeyPressed;
  CloseGraph;
End;  { LinesDrawing }

END.
```

```
{*************************************************************}
{*                                                           *}
{*    SWR Version 1.0                                        *}
{*    CopyRight (c) 1994 by Zulis Irawanto                   *}
{*    Teknik Perkapalan I T S                                *}
{*                                                           *}
{*************************************************************}

Unit WCForm;

{$N+,E+,S-,D-}

Interface

Uses WTVC;

Procedure CalcCForm (var Coordinate : RCoord;
                     var Cw, Cb, Cm, Cp : Single);

{ unit ini menghitung harga koefisien bentuk lambung kapal }
{ dengan menggunakan data titik-titik koordinat kapal      }

Implementation

Procedure CalcCForm;

Var
  I, J, K : byte;
  A, B    : single;

Begin
  With Coordinate do
  begin

    { Water plane coef. }

    Cw := 0;
    for I := 1 to nSt do
    begin
      K := I+(nWL-1)*nSt;
      Case I of
        1        : A := (X[2]-X[1])/2;
        2..nSt-1 : A := (X[I+1]-X[I-1])/2;
        nSt      : A := (X[nSt]-X[nSt-1])/2
      end;
      Cw := Cw+A*Y[K]
    end;
```

```
    { Midship coef. }

    Cm := 0;
    for J := 1 to nWL do
    begin
      K := Mid+(J-1)*nSt;
      Case J of
        1     : B := (Z[2]-Z[1])/2;
        nWL   : B := (Z[nWL]-Z[nWL-1])/2;
        else    B := (Z[J+1]-Z[J-1])/2
      end;
      Cm := Cm+B*Y[K]
    end;

    { Block coef. }

    Cb := 0;
    for I := 1 to nSt do
    begin
      Case I of
        1        : A := (X[2]-X[1])/2;
        2..nSt-1 : A := (X[I+1]-X[I-1])/2;
        nSt      : A := (X[nSt]-X[nSt-1])/2
      end;

      for J := 1 to nWL do
      begin
        Case J of
          1        : B := (Z[2]-Z[1])/2;
          2..nWL-1 : B := (Z[J+1]-Z[J-1])/2;
          nWL      : B := (Z[nWL]-Z[nWL-1])/2
        end;
        K := I+(J-1)*nSt;
        Cb := Cb+A*B*Y[K]
      end
    end;

  end; { Coordinate }

  { Prismatic Coef. }

  if Cm>1E-15 then Cp := Cb/Cm;

End;  { CalcCForm }

END.
```

```
{*************************************************************}
{*                                                           *}
{*    SWR Version 1.0                                        *}
{*    CopyRight (c) 1994 by Zulis Irawanto                   *}
{*    Teknik Perkapalan I T S                                *}
{*                                                           *}
{*************************************************************}

Unit WPrint;

{$N+,E+,S-,D-}

Interface

uses WTVC;

{   Unit ini bertugas mencetak data dari program:           }
{     * Data input dicetak dengan prosedur PrintInput      }
{     * Data Output dicetak dengan prosedur PrintOutput    }

procedure PrintInput (var MainDim: RMainDim; var Speed: RSpeed;
                      var Constraint: RConstraint);

procedure PrintOutPut (var Coordinate: RCoord; var MainDim: RMainDim;
                       var Speed: RSpeed; Cw, Cb, Cm, Cp, Cr: single);

Implementation

uses Printer;

procedure Enter(Line: byte);
var I: byte;
begin
  for I := 1 to Line do Writeln(Lst)
end;

procedure Space(Lenght: byte);
var I: byte;
begin
  for I := 1 to Lenght do Write(Lst,' ');
end;

procedure PrintOutPut;

var
 I, J, K : byte;

Begin  { PrintOutPut }
  Enter(3);
  Space(25); Writeln(Lst,'TABEL OFFSETS LAMBUNG KAPAL OPTIMUM');
  Space(25); Writeln(Lst,'═══════════════════════════════');
```

```
  Enter(1);
  Space(18);
  for I := 1 to 6 do
  begin
    Write(Lst,'WL',Coordinate.Z[I]:6:3);
    Space(2);
  end;

  Enter(2);
  for I := 1 to 13 do
  begin
    Space(5);
    Write(Lst,'Station',I:3);
    for J := 1 to 6 do
    begin
      K := I+(J-1)*13;
      Write(Lst,Coordinate.Y[K]:10:3)
    end;
    Enter(1);
  end;

  Enter(1);
  with MainDim do
  begin
    Space(5); Writeln(Lst,'Ukuran utama :');
    Space(5); Writeln(Lst,'■  T/Lpp :',(T/Lpp):7:3);
    Space(5); Writeln(Lst,'■  B/Lpp :',(B/Lpp):7:3);
  end;

  Enter(1);
  Space(5); Writeln(Lst,'Koefisien bentuk kapal :');
  Space(5); Writeln(Lst,'■  Cw  :',Cw:7:3);
  Space(5); Writeln(Lst,'■  Cb  :',Cb:7:3);
  Space(5); Writeln(Lst,'■  Cm  :',Cm:7:3);
  Space(5); Writeln(Lst,'■  Cp  :',Cp:7:3);

  Enter(1);
  Space(5); Writeln(Lst,'Kecepatan kapal:');
  Space(5); Write(Lst,'■  Fn  :');
  for I := 1 to Speed.nSpd do
    Write(Lst,Speed.Spd[I]:7:3);

  Enter(2);
  Space(5);
  if Speed.nSpd=1 then
    Writeln(Lst,'Koefisien tahanan gelombang :')
  else
    Writeln(Lst,'Koefisien tahanan gelombang rata-rata :');
  Space(5); Writeln(Lst,'■  Cr  :',Cr:7:3);

End;   { PrintOutPut }
```

```
procedure PrintInput;

var I, J, K : byte;

Begin  { PrintInput }

  Enter(3);
  with MainDim do
  begin
    Space(5); Writeln(Lst,'Ukuran utama :');
    Space(5); Writeln(Lst,'■  T/Lpp :',(T/Lpp):7:3);
    Space(5); Writeln(Lst,'■  B/Lpp :',(B/Lpp):7:3);
  end;

  Enter(1);
  with Speed do
  begin
    Space(5);
    Writeln(Lst,'Kecepatan kapal:');
    for I := 1 to nSpd do
    begin
      Space(5);
      Write(Lst,'■  Fn  :',Spd[I]:7:3);
      Space(5);
      if nSpd>1 then
        Writeln(Lst,'weighting factor :',Time[I]:7:3)
      Else
        Enter(1);
    end;
    Enter(1);
  end;

  with Constraint do
  begin
    Space(5); Writeln(Lst,'Constraints');
    if FixSt<>[] then
    begin
      Space(5);
      Write(Lst,'■  Station tetap       :');
      for I := 1 to nSt do
        if I in FixSt then Write(Lst,I:3);
    end;

    if FixWL<>[] then
    begin
      Enter(1);
      Space(5);
      Write(Lst,'■  Garis air tetap     :');
      for I := 1 to nWL do
        if I in FixWL then Write(Lst,I:3);
    end;
```

```
with Cw do
if Select<>[] then
begin
  Enter(1); Space(5);
  Write(Lst,'■ Koefisien Garis air :');
  if 1 in Select then write(Lst,Coef[1]:7:3,' ≤');
  Write(Lst,' Cw ');
  if 2 in Select then write(Lst,'≤',Coef[2]:7:3);
end;

With Cb do
if Select<>[] then
begin
  Enter(1); Space(5);
  Write(Lst,'■ Koefisien block     :');
  if 1 in Select then write(Lst,Coef[1]:7:3,' ≤');
  write(Lst,' Cb ');
  if 2 in Select then write(Lst,'≤',Coef[2]:7:3);
end;

With Cm do
if Select<>[] then
begin
  Enter(1); Space(5);
  Write(Lst,'■ Koefisien midship   :');
  if 1 in Select then write(Lst,Coef[1]:7:3,' ≤');
  Write(Lst,' Cm ');
  if 2 in Select then write(Lst,'≤',Coef[2]:7:3);
end;

with Cp do
if Select<>[] then
begin
  Enter(1); Space(5);
  Write(Lst,'■ Koefisien prismatik :');
  if 1 in Select then write(Lst,Coef[1]:7:3,' ≤');
  Write(Lst,' Cp ');
  if 2 in Select then write(Lst,'≤',Coef[2]:7:3);
end;

with WLAngle do
if SelectWL<>[] then
begin
  Enter(1); Space(5);
  Writeln(Lst,'■ Sudut garis air :');
  for I := 1 to nWL do
    if I in SelectWL then
    begin
      Space(10); Writeln(Lst,'WL',I:2,' :',Angle[I]:7:3);
    end;
end;
```

```
  With StAngle do
  if SelectSt<>[] then
  begin
    Enter(1); Space(5);
    Writeln(Lst,'■ Sudut station :');
    for I := 1 to nSt do
      if I in SelectSt then
      begin
        Space(10); Writeln(Lst,'Station',I:3,' :',Angle[I]:7:3);
      end;
  end;

  Enter(1); Space(5);
  Writeln(Lst,'■ Batas offsets :');
  for I := 1 to nSt do
    if not (I in FixSt) then
    begin
      Space(10); Write(Lst,'Station',I:3,' :');
      Space(2);  Write(Lst,MinOfs[1,I]:5:3,' ≤ Yi');
      if MaxOfs[1,I]<1E10 then
        Writeln(Lst,' ≤',MaxOfs[1,I]:6:3)
      else
        Enter(1);

      Space(24);
      if MinOfs[2,I]>1E-10 then
        Write(Lst,MinOfs[2,I]:5:3,' ≤ ');
      if (MinOfs[2,I]>1E-10) or (MaxOfs[2,I]<1E10) then
        Write(Lst,'Yi/Yi~');
      if MaxOfs[2,I]<1E10 then
        Writeln(Lst,' ≤',MaxOfs[2,I]:6:3)
      else
        Enter(1);
    end;

  Space(5); Writeln(Lst,'■ Catatan :');
  Space(10); Writeln(Lst,'Yi     = nilai offsets pada station i');
  Space(10);
  Writeln(Lst,'Yi/Yi~ = rasio antara offsets optimum dengan offsets');
  Space(19); Writeln(Lst,'pembanding pada station i');
end;  { Constraint }
End;    { PrintInput }

END.
```

```
{*************************************************************}
{*                                                           *}
{*    SWR Version 1.0                                        *}
{*    CopyRight (c) 1994 by Zulis Irawanto                   *}
{*    Teknik Perkapalan I T S                                *}
{*                                                           *}
{*************************************************************}

program SWRResource;

{$M 16384,8192,655360}

{   SWRResource adalah program untuk menghasilkan file      }
{   resource (SWR.RES) yang berisi menu dan semua window    }
{   dialog yang dipakai oleh program SWR                    }

uses WHelp, WType, WCmds, Drivers, Objects, Views, Menus,
     StdDlg,Dialogs, App;

var
  R       : tRect;
  Hint    : pStrListMaker;
  ResFile : tResourceFile;
  D       : pDialog;
  F       : pInputLine;
  C       : pCluster;
  L       : pLabel;
  I,J     : byte;
  S       : string[7];
  V       : PView;

procedure CreateHint;

begin
  Hint := New(PStrListMaker, Init(1000, 27));
  With Hint^ do
  begin
    Init (1000, 27);
    Put (hcAbout    ,'Nama program, versi program serta programer');
    Put (hcInput    ,'Input data yang diperlukan program');
    Put (hcMainDim ,'Ukuran utama kapal (Lpp,B,T)');
    Put (hcOfsPar   ,'Offsets lambung dari parrent ship');
    Put (hcSpeed    ,'Pada kecepatan berapa kapal akan dioptimumkan ?');
    Put (hcConstr   ,'Constraint yang dipakai di dalam optimisasi'
                     +' bentuk kapal');
    Put (hcFixOfs   ,'Garis air & station yang tidak diubah');
    Put (hcOfsBound,'Batas maksimum dan minimum offsets');
    Put (hcCForm    ,'Koefisien bentuk lambung (Cw,Cb,Cm,Cp)');
    Put (hcWLAngle ,'Sudut maksimum antara garis air & center line');
    Put (hcStAngle ,'Sudut minimum antara station & garis dasar');
    Put (hcPrintInput,'Cetak input data untuk lambung optimum');
```

```
    Put (hcRun      ,'Eksekusi program');
    Put (hcOutput   ,'Output data yang dihasilkan program');
    Put (hcOfsOpt   ,'Offsets lambung kapal optimum');
    Put (hcCResist ,'Koefisien tahanan gelombang');
    Put (hcCFormOpt,'Koefisien bentuk lambung optimum (Cw,Cb,Cm,Cp)');
    Put (hcSaveDat ,'Simpan data offsets optimum ke disk');
    Put (hcPrintOutput,'Cetak output data lambung optimum');
    Put (hcGraph    ,'Gambar lines plan kapal optimum');
    Put (hcGrShow   ,'Tampilkan gambar lines plan kapal optimum');
    Put (hcGrPrint ,'Mencetak gambar lines plan kapal optimum');
    Put (hcFiles    ,'Keluar dari program');
    Put (hcChDir    ,'Memilih directory yang aktif');
    Put (hcDosShel ,'Keluar dari program untuk sementara');
    Put (hcQuit,'Keluar dari program');
  end;
  ResFile.Put(Hint, 'Help');
  Dispose(Hint,Done);
end;

procedure CreateMenu;

begin
  R.Assign(0, 0, 80, 1);
  V := New(PMenuBar, Init(R, NewMenu(
    NewSubMenu('~'#240'~', hcSystem, NewMenu(
      NewItem('~I~nformasi program...', '', kbNoKey, cmAbout,
      hcAbout, nil)),
    NewSubmenu(' ~F~ile',hcFiles,Newmenu(
      NewItem('~U~bah directory...', '',kbNoKey, cmChDir, hcChDir,
      NewItem('~D~OS shell', '', kbNoKey, cmDosShell, hcDosShel,
      NewItem('E~x~it', 'Alt-X', kbAltX, cmExit, hcQuit, nil)))),
    NewSubmenu(' ~I~nput Data ',hcInput,Newmenu(
      NewItem('~U~kuran utama...', '',kbnokey,cmMainDim,hcMainDim,
      NewItem('~O~ffsets lambung...', '',kbnokey,cmOfsPar,hcOfsPar,
      NewItem('~K~ecepatan...', '',kbNoKey,cmSpeed,hcSpeed,
      NewSubmenu('~C~onstraints',hcConstr,Newmenu(
        NewItem('Offsets ~t~etap...', '',kbNoKey,cmFixOfs,hcFixOfs,
        NewItem('~B~atas offsets...', '',kbNoKey,cmOfsBound,hcOfsBound,
        NewItem('~K~oefisien bentuk...', '',kbNoKey,cmCForm,hcCForm,
        NewItem('Sudut garis ~a~ir...', '',kbNoKey,cmWLAngle,hcWLAngle,
        NewItem('Sudut ~s~tation...', '',kbNoKey,cmStAngle,hcStAngle,
        nil))))),
      NewLine(
      NewItem('~P~rint input data', '',kbNoKey,cmPrintInput,hcPrintInput,
      nil)))))),
    NewItem('~R~un', '',kbNoKey,cmRun,hcRun,
    NewSubmenu(' ~O~utput Data ',hcOutput,Newmenu(
      NewItem('~O~ffsets optimum...', '',kbNoKey,cmOfsOpt,hcOfsOpt,
      NewItem('Koefisien ~t~ahanan gelombang...', '',kbNoKey,
              cmCResist,hcCResist,
      NewItem('Koefisien ~b~entuk ...', '',kbNoKey,cmCFormOpt,hcCFormOpt,
```

```
      NewLine(
      NewItem('~S~impan data...', '', kbNoKey, cmSaveDat, hcSaveDat,
      NewItem('~P~rint output data', '', kbNoKey, cmPrintOutput,
              hcPrintOutPut, nil)))))),
    NewSubmenu('~G~rafik',hcGraph,Newmenu(
      NewItem('~T~ampilkan', '', kbNoKey, cmGrShow, hcGrShow,
      NewItem('~P~rint', '', kbNoKey, cmGrPrint, hcGrPrint,
      nil))), nil
  )))))))));
  ResFile.Put(V, 'MenuBar');
  Dispose(V, Done);
end;

procedure CreateFileReadDlg;

begin
  V := New(PFileDialog, Init('*.HUL', 'File Data',
    '~N~ama', fdOKButton + fdHelpButton, 1));
  V^.HelpCtx := hcSaveOpenFileDlg;
  ResFile.Put(V, 'FileReadDlg');
  Dispose(V, Done);
end;

procedure CreateFileSaveDlg;

begin
  V := New(PFileDialog, Init('*.HUL', 'Menyimpan file',
       '~N~ama', fdOKButton + fdHelpButton, 2));
  V^.HelpCtx := hcSaveOpenFileDlg;
  ResFile.Put(V, 'FileSaveDlg');
  Dispose(V, Done);
end;

procedure CreateChDirDlg;

begin
  D := New(PChDirDialog, Init(cdNormal+cdHelpButton+cdNoLoadDir,101));
  D^.HelpCtx := hcChDirDlg;
  ResFile.Put(D, 'ChDirDlg');
  Dispose(D, Done);
end;

procedure CreateXZCoordDlg;

begin
  R.Assign(4, 2, 50, 21);
  D := New(PDialog, Init(R, 'Hull Coordinate'));
  with D^ do
  begin
    R.Assign(2,2,20,3);
    Insert(New(PStaticText, Init(R, 'Posisi garis air :')));
```

```
  for I := 1 to 6 do
  begin
    Case I of
      1..3  : R.Assign(3, 3+I, 7, 4+I);
      4..6  : R.Assign(18, I, 22, I+1);
    end;
    str(I, S);
    Insert(New(PStaticText, Init(R, 'WL '+S)));
  end;
  for I := 1 to 6 do
  begin
    Case I of
      1..3  : R.Assign(8, 3+I, 15, 4+I);
      4..6  : R.Assign(24, I, 31, I+1);
    end;
    Insert(New(pRealInputLine, Init(R, 7)))
  end;

  R.Assign(2,8,20,9);
  Insert(New(PStaticText, Init(R, 'Posisi station :')));

  for I := 1 to 13 do
  begin
    Case I of
      1..7  : R.Assign(3, 9+I, 7, 10+I);
      8..13 : R.Assign(18, I+2, 23, I+3);
    end;
    str(I, S);
    Insert(New(PStaticText, Init(R, 'St '+S)));
  end;
  for I := 1 to 13 do
  begin
    Case I of
      1..7  : R.Assign(8,9+I,15,10+I);
      8..13 : R.Assign(24,I+2,31,I+3);
    end;
    Insert(New(pRealInputLine, Init(R, 7)))
  end;

  R.Assign(33, 4, 44, 6);
  Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
  R.Assign(33, 7, 44, 9);
  Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
  R.Assign(33, 10, 44, 12);
  Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
  SelectNext(False);
 end;
 D^.HelpCtx := hcXZCoord;
 ResFile.Put(D, 'XZCoord');
 Dispose(D,Done);
end;
```

```
procedure CreateYCoordDlg;
begin
  R.Assign(3,1,78,21);
  D := New(PDialog, Init(R, 'Hull Coordinate'));
  with D^ do
  begin
    for I := 1 to 13 do
    begin
      str(I, S);
      R.Assign(2, 3+I, 10, 4+I);
      Insert(New(PStaticText, Init(R, 'Sta. '+ S)));
    end;
    R.Assign(12, 2, 70, 3);
    Insert(New(PStaticText, Init(R, 'WL  1    WL  2   '+
              ' WL  3    WL  4    WL  5    WL  6   ')));
    for I := 1 to 6 do
      for J := 1 to 13 do
      begin
        R.Assign(2+9*I,3+J,9+9*I,4+J);
        Insert(New(pRealInputLine, Init(R, 7)));
      end;
    R.Assign(63, 4, 73, 6);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(63, 7, 73, 9);
    Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
    R.Assign(63, 10, 73, 12);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(False);
  end;
  D^.HelpCtx := hcYCoord;
  ResFile.Put(D,'YCoord');
  Dispose(D,Done);
end;

procedure CreateAboutDlg;
begin
  R.Assign(0, 0, 38, 14);
  D := New(PDialog, Init(R, ''));
  with D^ do
  begin
    Options := Options or ofCentered;
    R.Assign(7, 3, 30, 12);
    Insert(New(PStaticText, Init(R,'SWR Version 1.0'#13#13+
    'Optimal Ship Form for'#13+'Minimum Wave Resistance'#13#13+
    'Copyright (c) 1994'#13+'by Zulis Irawanto'#13+
    'Teknik Perkapalan - ITS')));
  end;
  D^.HelpCtx := hcProgInfo;
  ResFile.Put(D,'About');
  Dispose(D, Done);
end;
```

```
procedure CreateMainDimDlg;
begin
  R.Assign (0,0,36,13);
  D := New(PDialog, Init(R, 'Ukuran Utama'));
  with   D^ do
  begin
      Options := Options or ofCentered;
      R.Assign(23,3,33,4);
      F := New(PRealInputLine, Init(R, 7));
      Insert(F);
      R.Assign(3,3,23,4);
      Insert(New(PLabel, Init(R, 'Panjang  ( ~L~pp ) :', F)));

      R.Assign(23,5,33,6);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);
      R.Assign(3,5,23,6);
      Insert(New(PLabel, Init(R, 'Lebar    ( ~B~  ) :', F)));

      R.Assign(23,7,33,8);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);
      R.Assign(3,7,23,8);
      Insert(New(PLabel, Init(R, 'Sarat    ( ~T~  ) :', F)));

      R.Assign(2,10,12,12);
      Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
      R.Assign(13, 10 , 23, 12);
      Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
      R.Assign(24, 10, 34, 12);
      Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
      SelectNext(False);
    end;
  D^.HelpCtx := hcMainDimDlg;
  ResFile.Put(D, 'MainDim');
  Dispose(D, Done);
end;

procedure CreateOfsParDlg;
begin
  R.Assign(0,0,31,13);
  D := New(PDialog, Init(R, 'Parrent offsets'));
  with   D^ do
  begin
    Options := Options or ofCentered;
    R.Assign(4,3,28,6);
    C := New(PRadioButtons, Init(R,
         NewSItem('~S~eries60 Block60',
         NewSItem('Data dari ~F~ile',
         NewSItem('Data dari ~K~eyboard', nil)))));
    Insert(C);
```

```
    R.Assign(2, 8, 14, 10);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(16, 8, 28, 10);
    Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
    R.Assign(9, 10, 21, 12);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(false);
  end;
  D^.HelpCtx := hcOfsParDlg;
  ResFile.Put(D, 'OfsPar');
  Dispose(D,Done);
end;

procedure CreateFixOfsDlg;

begin
  R.Assign(0,0,52,18);
  D := New(PDialog, Init(R, 'Constraint'));
  with D^ do
  begin
    Options := Options or ofCentered;
    R.Assign(2,3,34,6);
    C := New(PCheckBoxes, Init(R,
      NewSItem('WL 1 (BL)', NewSItem('WL 2',
      NewSItem('WL 3', NewSItem('WL 4',
      NewSItem('WL 5', NewSItem('WL 6 (DWL)', nil)))))));
    Insert(C);
    R.Assign(2,2,24,3);
    Insert(New(PLabel, Init(R, '~G~aris air yang tetap', C)));

    R.Assign(2,8,50,13);
    C := New(PCheckBoxes, Init(R,
      NewSItem('FP (St.1)', NewSItem('Station 2',
      NewSItem('Station 3', NewSItem('Station 4',
      NewSItem('Station 5', NewSItem('Station 6',
      NewSItem('Station 7', NewSItem('Station 8',
      NewSItem('Station 9', NewSItem('Station 10',
      NewSItem('Station 11', NewSItem('Station 12',
      NewSItem('AP (St.13)', nil)))))))))))));
    Insert(C);
    R.Assign(2, 7, 22, 8);
    Insert(New(PLabel, Init(R, '~S~tation yang tetap', C)));

    R.Assign(10, 15, 22, 17);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(24, 15, 36, 17);
    Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
    R.Assign(38, 15, 50, 17);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(False);
  end;
```

```
  D^.HelpCtx := hcFixOfsDlg;
  ResFile.Put(D, 'FixOfs');
  Dispose(D, Done);
end;

procedure CreateSpeedDlg;
begin
  R.Assign(0, 0, 55, 19);
  D := New(PDialog, Init(R, 'Speed'));
  with D^ do
  begin
    Options := Options or ofCentered;
    R.Assign(20,3,23,4);
    F := New(pRealInputLine, Init(R, 1));
    Insert(F);

    R.Assign(1,3,20,4);
    Insert(New(PLabel, Init(R,'~J~umlah kecepatan:',F)));
    R.Assign(26,3,36,4);
    Insert(New(PStaticText, Init(R,'Kecepatan')));
    R.Assign(37,3,53,4);
    Insert(New(PStaticText, Init(R,'Weighting Factor')));

    for I := 1 to 6 do
    begin
      R.Assign(26,3+2*I,35,4+2*I);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);
      R.Assign(23,3+2*I,26,4+2*I);
      Insert(New(PLabel, Init(R, '~'+char(i+48)+'~', F)));
    end;
    for I := 1 to 6 do
    begin
      R.Assign(41,3+2*I,50,4+2*I);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);
      R.Assign(38,3+2*I,40,4+2*I);
      Insert(New(PLabel, Init(R, '~'+char(i+96)+'~', F)));
    end;
    R.Assign(2, 9, 14, 11);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(2, 12, 14, 14);
    Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
    R.Assign(2, 15, 14, 17);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(False)
  end;
  D^.HelpCtx := hcSpeedDlg;
  ResFile.Put(D, 'Speed');
  Dispose(D, Done);
end;
```

```
procedure CreateCFormDlg;

begin
  R.Assign(0, 0, 35, 14);
  D := New(PDialog, Init(R, 'Constraint'));
  with D^ do
  begin
    Options := Options or ofCentered;
    for I := 1 to 4 do
    begin
      R.Assign(4, 2*I, 14, 2*I+1);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);

      R.Assign(22, 2*I, 32, 2*I+1);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);

      R.Assign(15, 2*I, 21, 2*I+1);
      case I of
        1: Insert(New(PStaticText, Init(R,'≤ Cw ≤')));
        2: Insert(New(PStaticText, Init(R,'≤ Cb ≤')));
        3: Insert(New(PStaticText, Init(R,'≤ Cm ≤')));
        4: Insert(New(PStaticText, Init(R,'≤ Cp ≤')));
      end;
    end;

    R.Assign(1, 11, 11, 13);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(12, 11, 22, 13);
    Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
    R.Assign(23, 11, 33, 13);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(False)
  end;

  D^.HelpCtx := hcCFormDlg;
  ResFile.Put(D, 'CForm');
  Dispose(D, Done);
end;

procedure CreateWLAngleDlg;

begin
  R.Assign(0, 0, 38, 15);
  D := New(PDialog, Init(R, 'Constraint'));
  with D^ do
  begin
    Options := Options or ofCentered;
    R.Assign(3,2,30,3);
    Insert(New(PStaticText, Init(R,'Sudut garis air maksimum:')));
```

```
    for I := 1 to 6 do
    begin
      case I of
        1..3 : begin
                 R.Assign(8, 2*I+2, 17, 2*I+3);
                 F := New(pRealInputLine, Init(R, 7));
                 Insert(F);
                 R.Assign(2, 2*I+2, 7, 2*I+3);
                 Str(I,S); S := '~'+S+'~';
                 Insert(New(PLabel, Init(R, ('WL '+S), F)));
               end;
        4..6 : begin
                 R.Assign(26, 2*I-4, 35, 2*I-3);
                 F := New(pRealInputLine, Init(R, 7));
                 Insert(F);
                 R.Assign(20, 2*I-4, 25, 2*I-3);
                 Str(I,S); S := '~'+S+'~';
                 Insert(New(PLabel, Init(R, ('WL '+S), F)));
               end;
      end;
    end;

    R.Assign(1, 11, 11, 13);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(13, 11, 23, 13);
    Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
    R.Assign(25, 11, 35, 13);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(False)
  end;

  D^.HelpCtx := hcWLAngleDlg;
  ResFile.Put(D, 'WLAngle');
  Dispose(D, Done);
end;

procedure CreateStAngleDlg;

begin
  R.Assign(0, 0, 53, 19);
  D := New(PDialog, Init(R, 'Constraint'));
  with D^ do
  begin
    Options := Options or ofCentered;
    R.Assign(3,2,29,3);
    Insert(New(PStaticText, Init(R,'Sudut station maksimum:')));

    for I := 1 to 13 do
    begin
      case I of
        1: S := '~F~P';
```

```
        2..9: begin
                Str(I,S);
                S := 'St.~'+S+'~';
              end;
        10 : S := 'St.1~0~';
        11 : S := 'St.1~1~';
        12 : S := '~S~t.12';
        13 : S := '~A~P';
      end;
      case I of
        1..7 : begin
                 R.Assign(9, 2*I+2, 18, 2*I+3);
                 F := New(pRealInputLine, Init(R, 7));
                 Insert(F);
                 R.Assign(2, 2*I+2, 9, 2*I+3);
                 Insert(New(PLabel, Init(R, (S), F)));
               end;
        8..13 : begin
                 R.Assign(28, 2*I-12, 37, 2*I-11);
                 F := New(pRealInputLine, Init(R, 7));
                 Insert(F);
                 R.Assign(21, 2*I-12, 28, 2*I-11);
                 Insert(New(PLabel, Init(R, (S), F)));
               end;
      end;
    end;

    R.Assign(39, 4, 51, 6);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(39, 7, 51, 9);
    Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
    R.Assign(39, 10, 51, 12);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(False)
  end;
  D^.HelpCtx := hcStAngleDlg;
  ResFile.Put(D, 'StAngle');
  Dispose(D, Done);
end;

procedure CreateOfsBoundDlg;

begin
  R.Assign(3, 1, 75, 22);
  D := New(PDialog, Init(R, 'Constraint'));
  with D^ do
  begin
    R.Assign(2,2,24,3);
    Insert(New(PStaticText, Init(R,'Min ≤ Y optimum ≤ Max')));
    R.Assign(2,4,24,5);
    Insert(New(PStaticText, Init(R,'St.     Min       Max')));
```

```
    for I := 1 to 13 do
    begin
      R.Assign(2,I+5,5,I+6);
      Str(I,S);
      Insert(New(PStaticText, Init(R,S)));

      R.Assign(6,I+5,16,I+6);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);

      R.Assign(17,I+5,27,I+6);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);
    end;

    R.Assign(30,2,68,3);
    Insert(New(PStaticText, Init(R,
       'Min ≤ Y optimum / Y parrent ≤ Max')));
    R.Assign(30,4,52,5);
    Insert(New(PStaticText, Init(R,'St.     Min       Max')));

    for I := 1 to 13 do
    begin
      R.Assign(30,I+5,32,I+6);
      Str(I,S);
      Insert(New(PStaticText, Init(R,S)));

      R.Assign(34,I+5,44,I+6);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);

      R.Assign(45,I+5,55,I+6);
      F := New(pRealInputLine, Init(R, 7));
      Insert(F);
    end;
    R.Assign(56,16,71,17);
    Insert(New(PStaticText, Init(R,'Y= hull offsets')));

    R.Assign(57, 6, 69, 8);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(57, 9, 69, 11);
    Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
    R.Assign(57, 12, 69, 14);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(False)
  end;

  D^.HelpCtx := hcOfsBoundDlg;
  ResFile.Put(D, 'OfsBound');
  Dispose(D,Done);
end;
```

```
procedure CreateOfsOptimumDlg;

begin
  R.Assign(3,1,78,21);
  D := New(PDialog, Init(R, 'Offsets Lambung Optimum'));
  with D^ do
  begin
    for I := 1 to 13 do
    begin
      str(I, S);
      R.Assign(2, 3+I, 10, 4+I);
      Insert(New(PStaticText, Init(R, 'Sta. '+ S)));
    end;
    R.Assign(12, 2, 70, 3);
    Insert(New(PStaticText, Init(R, 'WL  1    WL  2    '+
    ' WL  3    WL  4    WL  5    WL  6    ')));

    for I := 1 to 6 do
      for J := 1 to 13 do
      begin
        R.Assign(2+9*I,3+J,9+9*I,4+J);
        flags := flags and Not ofSelectable;
        Insert(New(pRealInputLine, Init(R, 7)));
      end;

    R.Assign(63, 4, 73, 6);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(63, 7, 73, 9);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(False);
  end;
  D^.HelpCtx := hcOfsOptDlg;
  ResFile.Put(D,'OfsOpt');
  Dispose(D,Done);
end;

procedure CreateLinesDlg;

begin
  R.Assign(0,0,35,15);
  D := New(PDialog, Init(R,'Rencana Garis'));
  with   D^ do
  begin
    Options := Options or ofCentered;
    R.Assign(3,3,20,5);
    C := New(PRadioButtons, Init(R,
         NewSItem('~B~ody plan',
         NewSItem('~W~ater plan', nil))));
    Insert(C);
    R.Assign(3,2,25,3);
    Insert(New(PLabel, Init(R, 'Lines plan', C)));
```

```
    R.Assign(3,7,28,9);
    C := New(PRadioButtons, Init(R,
         NewSItem('~S~emua',
         NewSItem('~D~epan',
         NewSItem('Bela~k~ang', nil)))));
    Insert(C);
    R.Assign(3,6,28,7);
    Insert(New(PLabel, Init(R, 'Seksi yang ditampilkan', C)));
    R.Assign(1, 11, 12, 13);
    Insert(New(PButton, Init(R, '~O~K', cmOk, bfDefault)));
    R.Assign(12, 11, 22, 13);
    Insert(New(PButton, Init(R, 'Cancel', cmCancel, bfNormal)));
    R.Assign(23, 11, 33, 13);
    Insert(New(PButton, Init(R, 'Help', cmHelp, bfNormal)));
    SelectNext(false);
  end;
  D^.HelpCtx := hcLinesDlg;
  ResFile.Put(D, 'Lines');
  Dispose(D,Done);
end;

BEGIN  { SWRResource }
  ResFile.Init (New (PBufStream, Init ('SWR.RES', stCreate, 2500)));
  RegisterObjects;
  RegisterType (RStrListMaker);
  RegisterType (RRealInputLine);
  RegisterViews;
  RegisterMenus;
  RegisterDialogs;
  RegisterStdDlg;

  CreateHint;
  CreateMenu;
  CreateAboutDlg;
  CreateMainDimDlg;
  CreateSpeedDlg;
  CreateOfsParDlg;
  CreateXZCoordDlg;
  CreateYCoordDlg;
  CreateFixOfsDlg;
  CreateOfsBoundDlg;
  CreateCFormDlg;
  CreateWLAngleDlg;
  CreateStAngleDlg;
  CreateFileReadDlg;
  CreateFileSaveDlg;
  CreateChDirDlg;
  CreateOfsOptimumDlg;
  CreateLinesDlg;
  ResFile.Done;
END.
```